USACERL ADP Report P-91/45
August 1991

US Army Corps
of Engineers
Construction Engineering
Research Laboratory

Maintenance Resource Prediction Model Summary System (MRPMSS) Programmer's Manual

by
Edgar S. Neely
Robert D. Neathammer
Bill Wang

Maintenance Resource Prediction Models (MRPMs) are a set of models that run on various computer systems to assist Army managers to plan and program maintenance resources, based on the anticipated resource requirements of actual installation facilities, for prediction periods of 1 to 10 years.

This manual provides system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system.

This data base and computer system is presently used by U.S. Army Corps of Engineers (USACE) designers at district and installation levels, and by resource programmers at the USACE Headquarters, Army Major Command, and installation levels. MRPMs may also prove useful to other Government agencies and to the private sector.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1991	Final

4. TITLE AND SUBTITLE
Maintenance Resource Prediction Model Summary System (MRPMSS) Programmer's Manual

5. FUNDING NUMBERS
RDTE dated 1980
REIM from
1984 to 1989

6. AUTHOR(S)
Edgar S. Neely, Robert D. Neathammer, and Bill Wang

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Construction Engineering Research Laboratory (USACERL)
PO Box 9005
Champaign, IL 61826-9005

8. PERFORMING ORGANIZATION REPORT NUMBER
ADP P-91/45

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
COMMANDER
U.S. Army Engineering and Housing Support Center
ATTN: CEHSC-FM-R
Fort Belvoir, VA 22060

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Copies are available from the National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Maintenance Resource Prediction Models (MRPMs) are a set of models that run on various computer systems to assist Army managers to plan and program maintenance resources, based on the anticipated resource requirements of actual installation facilities, for prediction periods of 1 to 10 years.

This manual provides system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system.

This data base and computer system is presently used by U.S. Army Corps of Engineers (USACE) designers at district and installation levels, and by resource programmers at the USACE Headquarters, Army Major Command, and installation levels. MRPMs may also prove useful to other Government agencies and to the private sector.

14. SUBJECT TERMS
Maintenance Resource Prediction Models (MRPM)
programming manuals

15. NUMBER OF PAGES
80

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

FOREWORD

This research was conducted for the Office of the Chief of Engineers (OCE), under various RDTE and FAD funding documents. Work began under RDTE in 1980 and continued in reimbursable projects from 1984 through 1989. The technical monitors were, for the direct-funded part, Dr. Larry Schindler, CEMP-ECE-G, and at the termination of the project, Mr. Ed Davis, CEHSC-FM-R; and for the reimbursable part, Ms. Val Corbridge, DAEN-ZCP-B.

The work was performed by the Facility Systems Division (FS), U.S. Army Construction Engineering Research Laboratory (USACERL). Dr. Edgar S. Neely was the USACERL principal investigator. Dr. Michael J. O'Connor is Chief, USACERL-FS. The USACERL technical editor was Mr. William J. Wolfe, Information Management Office.

COL Everett R. Thomas is Commander and Director of USACERL, and Dr. L.R. Shaffer is Technical Director.

CONTENTS

		Page
	SF 298	1
	FOREWORD	2
1	INTRODUCTION	5
2	LEARNING THE FUNCTIONS	6
3	PROGRAM FLOW	7
4	STANDARD SUBROUTINES	16
5	STANDARD COMMON BLOCKS	53
6	STANDARD PROGRAMMING PACKAGES	69
7	REQUIRED PROGRAMMING PRACTICES	70
8	MANAGEMENT PROCEDURES	74
9	RESOURCES	76
	Supervision	
	Training	
	Hotline	
	PC System Maintenance	
	Newsletter	
	Cost Summary	
	DISTRIBUTION	

MAINTENANCE RESOURCE PREDICTION MODEL SUMMARY SYSTEM (MRPMSS) PROGRAMMER'S MANUAL

1 INTRODUCTION

The primary purpose of this manual is to provide the system programmers with a comprehensive description of each procedure required to learn, operate, and maintain the personal computer MRPM summary system. Chapter 2 describes the most efficient method for learning the functions and organization of the MRPMSS system. Chapter 3 defines the program flow from subroutine to subroutine. Chapter 4 contains the description of all standard or common subroutines that must be used by all programmers when writing new code or modifying existing code. Chapter 5 describes all standard common blocks that must be used when programming. Chapter 6 contains a list of standard programming packages used in the MRPMSS system. Chapter 7 describes the procedures to be followed during system maintenance and operation. Chapter 8 describes the overall management procedures required to operate the system. Chapter 9 describes resources required to support the MRPMSS system.

2 LEARNING THE FUNCTIONS

The first and most important step is to train the maintainer as a functional user of the system. Give the new person a user's manual and access to the MRPMSS system on a personal computer (PC). Have the person read the manual, learn the system, and write down all questions. Do not give the person any verbal description of the system. All information should be contained in the user's manual.

Revise the user's manual as needed using the programmer's questions. If the new person had the question, it is probable others will also. This method constantly improves both user and system documentation.

The user's manual is a self-teach document. The learning process takes approximately 1 week and should be very smooth and efficient. Once the maintainer knows the functions, this programmer's manual can be used.

3 PROGRAM FLOW

This chapter presents the flow of the program by functional use. Table 1 contains the functional use as displayed on the screen, the program named, and the files as tables accessed. Table 2 contains the same information ordered by program name. Table 3 contains the information ordered by file name.

Table 1

MRPM Files
Ordered as Shown on the Screen

Function	Program	Data Files
INSTMENU MENU		
INSTMENU .	INSTMEMU.EXE*	INSTMENU.TB1
MAIN MENU		
MRPM .	MRPMSSS.EXE	MRPMTEMP.DAT MRPMTEMP.BAT
BASIC INFORMATION		
GENERAL INFORMATION		
Organization Chart .	ORGCHRT.EXE	ORGFGC.XDB
RMF Factors .	RMF-FACT EXE	RMF-FACT.XDB INSTINFO DAT
F4C Conversion Codes .	F4CAMS.EXE	AMSF4C.XDB
Report Periods .	RP-DATES.EXE	INSTINFO.DAT ORGFGC.XDB
Unit Cost Factors	UNC-FACT.EXE	UNC-FACT.XDB UNCDSC.XDB
U.S. Factor Description Editor	COST-DES.EXE	UNCDSC.XDB
AMS Description Editor .	AMS-EDIT.EXE	AMSDSC.XDB
Area Identification .	AREA_TAB.EXE	AREA_TAB.XDB INSTINFO.DAT SUB-TAB.XDB
Subinstallation Identification.	SUB-TAB.EXE	INSTINFO.DAT SUB-TAB. XDB

*The listing of programs is in menu order. For each MRPM function there is a corresponding program file name and data files accessed.

Table 1 (Cont'd)

Function	Program	Data Files
Lan unit cost graph .	CHART.EXE	
FACILITY RESOURCE DATA		
F4C Resource Description Table	EDF4CS.EXE	DES-BTSM.XDB DES-TASK.XDB DES-TRWD.XDB F4CDESC.XDB F4C-YEAR.XDB INSTINFO.DAT
Trade and Costs .	TRD-EDIT.EXE	INSTINFO.DAT TRDCOSTS.DAT
Total/Partial Summary Tasks .	DES-EDT3.EXE	BTSMtdpd.XDB BT-PASS.DAT CLASLST.DAT DES-BTSM.XDB INSTINFO.DAT TRDCOSTS.DAT TRWDti--.XDB VALLIST.DAT
F4C Description Editor .	F4C-EDIT.EXE	F4CDESC.XDB
FACILITY INFORMATION		
RESOURCE CALCULATION .	FACASS.EXE	AMSF4C.XDB BTSMtdpd.XDB CTODsecq.XDB F4C-YEAR.XDB FACILITY.XDB INSTINFO.DAT RMF-FACT.XDB RSMTTOTL.XDB RSMYsecq.XDB SCMIDid.DAT TASKtigi.XDB TRAVTIME.DAT TRDCOSTS.DAT TREEsecq.DAT UNC-FACT.XDB
	(MENUSS sub) (FIOPEN sub) (OCECAL sub) (SQFCAL sub) (STDCAL sub) (UNCCAS sub)	
DISPLAY RESOURCES		
Display Facility Totals .	RSMTDPS.EXE	FACILITY.XDB INSTINFO.DAT RSMTTOTL.XDB SYLCHART.ASC
	(RSMTREPT.EXE)	
GENERAL INFORMATION .	FA-XEDIS.EXE	AMSF4C.XDB

Table 1 (Cont'd)

Function	Program	Data Files
		AREA_TAB.XDB
		FACILITY.XDB
		FFPROF.XDB
		F4CDESC.XDB
		INSTINFO.DAT
		SCMDEF.XDB
		SUB-TAB.XDB
		WP-DESC DAT
FACILITY REPORTS		
F4C/AMS Organizational Summary	AMSF4CRP.EXE	AMSF4C.XDB
		FACILITY.XDB
		INSTINFO.DAT
		RSMTTOTL.XDB
Facility Totals Report .	RSMTREPS.EXE	FACILITY.XDB
		INSTINFO DAT
		RSMTTOTL.XDB
Unconstrained Reqrmnts Reports		
URR Editor .	URREDIT.EXE	AMSDSC.XDB
		INSTINFO.DAT
		URR.XDB
		ZZZZZZZZ.URR
Escalation Factor Editor .	URRAPR.EXE	INSTINFO.DAT
		UNC-FACT.XDB
Produce Reports .	URRCOP.EXE	AMSDSC.XDB
		AMSF4C.XDB
		FACILITY.XDB
		INSTINFO.DAT
		RSMTTOTL.XDB
		URR.XDB
		URRAPR.XDB
View and Print Report Files .	VIEWRP.EXE	ZZZZZZZZ.FTR
		ZZZZZZZZ.OSR
		ZZZZZZZZ.UCR
		ZZZZZZZZ.URA
		ZZZZZZZZ.URC
MODEL FACILITY		
Standard .	DATFAC.EXE	FACILITY.XDB
National Guard .	NGDFAC.EXE	FACILITY.XDB
Non Army Organizations .	NOAFAC.EXE	FACILITY.XDB
V Corps .	VCORPS.EXE	FACILITY.XDB
DELETE RESOURCE TOTAL FILE .	DRESTOS.EXE	RSMTTOTL.XDB
COMBINE FACILITY TOTALS FILES .	COM-FAC.EXE	
LAN DISPLAY GRAPH .	CHART.EXE	

Table 2

MRPM Files
Ordered by Program Name

Function	Program	Data Files
AMS Description Editor	AMS-EDIT.EXE*	AMSDSC.XDB
F4C/AMS Organizational Summary	AMSF4CRP.EXE	AMSF4C.XDB FACILITY.XDB INSTINFO.DAT RSMTTOTL.XDB
Area Identification	AREA_TAB.EXE	AREA_TAB.XDB
LAN Unit Cost/Display Graph	CHART.EXE	
COMBINE FACILITY TOTALS FILES	COM-FAC.EXE	
U.S. Factor Description Editor	COST-DES.EXE	UNCDSC.XDB
Standard		FACILITY XDB
Total/Partial Summary Tasks	DES-EDT3.EXE	BTSMtdpd.XDB BT-PASS.DAT CLASLST.DAT DES-BTSM.XDB INSTINFO.DAT TRDCOSTS.DAT TRWDti--.XDB VALLIST.DAT
DELETE RESOURCE TOTAL FILE	DRESTOS.EXE	RSMTTOTL.XDB
F4C Resource Description Table	EDF4CS.EXE	DES-BTSM.XDB DES-TASK.XDB DES-TRWD.XDB F4CDESC.XDB F4C-YEAR.XDB INSTINFO.DAT
F4C Conversion Codes	F4CAMS.EXE	AMSF4C.XDB
F4C Description Editor	F4C-EDIT.EXE	F4CDESC.XDB
RESOURCE CALCULATION	FACASS.EXE	AMSF4C.XDB BTSMtdpd.XDB CTODsecq.XDB F4C-YEAR.XDB FACILITY.XDB INSTINFO.DAT RMF-FACT.XDB RSMTTOTL.XDB RSMYsecq.XDB SCMIDid. DAT TASKtiqi.XDB

*The listing of programs is in alphabetical order. For each program there is a corresponding MRPM function and data files accessed.

Table 2 (Cont'd)

Function	Program	Data Files
		TRAVTIME.DAT TRDCOSTS.DAT TREEsecq.DAT UNC-FACT.XDB
GENERAL INFORMATION	FA-XEDIS.EXE	AMSF4C.XDB AREA_TAB.XDB FACILITY.XDB FFPROF.XDB F4CDESC.XDB INSTINFO.DAT SCMDEF.XDB SUB-TAB.XDB WP-DESC.DAT
INSTMENU	INSTMENU.EXE	INSTMENU.TBL
MRPM	MRPMSS.EXE	MRPMTEMP.DAT MRPMTEMP.BAT
National Guard	NGDFAC.EXE	FACILITY.XDB
Non Army Organizations	NOAFAC.EXE	FACILITY.XDB
Organization Chart	ORGCHRT.EXE	ORGFGC.XDB
RMF Factors	RMF-FACT.EXE	INSTINFO.DAT RMF-FACT.XDB
Report Periods	RP-DATES.EXE	INSTINFO.DAT ORGCHRT.XDB
Display Facility Totals	RSMTPDS.EXE	FACILITY.XDB INSTINFO.DAT RSMTTOTL.XDB SYLCHART.ASC
Facility Totals Report	RSMTREPS.EXE	FACILITY.XDB INSTINFO.DAT RSMTTOTL.XDB
Subinstallation Identification	SUB-TAB.EXE	INSTINFO.DAT SUB-TAB.XDB
Trade and Costs	TRD-EDIT.EXE	INSTINFO.DAT TRDCOSTS.DAT
Unit Cost Factors	UNC-FACT.EXE	UNC-FACT.XDB UNCDSC.XDB
Escalation Factor Editor	URRAPR.EXE	INSTINFO.DAT UNC-FACT.XDB
Produce Reports	URRCOP.EXE	AMSDSC.XDB AMSF4C.XDB FACILITY.XDB INSTINFO.DAT RSMTTOTL.XDB

Table 2 (Cont'd)

Function	Program	Data Files
		URR.XDB URRAPR.XDB
URR Editor	URREDIT.EXE	AMSDSC.XDB INSTINFO.DAT URR.XDB ZZZZZZZ. URR
V Corps	VCORPS.EXE	FACILITY.XDB
View and Print Report Files	VIEWRP.EXE	ZZZZZZZZ.FTR ZZZZZZZZ.OSR ZZZZZZZZ.UCR ZZZZZZZZ.URA ZZZZZZZZ.URC

Table 3

MRPM Program Files Ordered by Data File Name

Program	Data Files
AMS-EDIT.EXE	AMSDSC.XDB*
URRCOP.EXE	
URREDIT.EXE	
AMSF4CRP.EXE	AMSF4C.XDB
F4CAMS.EXE	
FACASS.EXE	
FA-XEDIS.EXE	
URRCOP.EXE	
AREA_TAB.EXE	AREA-TAB.XDB
FA-XEDIS.EXE	
DES-EDT3.EXE	BTSMtdpd.XDB
FACASS.EXE	
DES-EDT3.EXE	BT-PASS.DAT
DES-EDT3.EXE	CLASLST.DAT
FACASS.EXE	CTODsecq.XDB
DES-EDT3.EXE	DES-BTSM.XDB
EDF4C.EXE	
EDF4CS.EXE	DES-TASK.XDB
EDF4CS.EXE	DES-TRWD.XDB
EDF4CS.EXE	F4CDESC.XDB
F4C-EDIT.EXE	
FA-XEDIS.EXE	
EDF4CS.EXE	F4C-YEAR.XDB
FACASS.EXE	
AMSF4CRP.EXE	FACILITY.XDB
DATFAC.EXE	
FA-XEDIS.EXE	
FACASS.EXE	
NGDFAC.EXE	
NOAFAC.EXE	
RSMTDPS.EXE	
RSMTREPS.EXE	
URRCOP.EXE	
VCORPS.EXE	

*The listing of data file name is in alphabetical order. Each data file has corresponding execut files.

Table 3 (Cont'd)

Program	Data Files
FA-XEDIS.EXE	FFPROF.XDB
AMSF4CRP.EXE DES-EDT3.EXE EDF4CS.EXE FACASS.EXE FA-XEDIS.EXE RMF-FACT.EXE RP-DATES.EXE RSMTDPS.EXE RSMTREPS.EXE SUB-TAB.EXE TRD-EDIT.EXE URRAPR.EXE URRCOP.EXE URREDIT.EXE	INSTINFO.DAT
INSTMEMU.EXE	INSTMENU.TB1
MRPMSSS.EXE	MRPMTEMP.BAT
MRPMSSS.EXE	MRPMTEMP.DAT
ORGCHRT.EXE RP-DATES.EXE	ORGFGC.XDB
FACASS.EXE RMF-FACT.EXE	RMF-FACT.XDB
AMSF4CRP.EXE DRESTOS.EXE FACASS.EXE RSMTDPS.EXE RSMTREPS.EXE URRCOP.EXE	RSMTTOTL.XDB
FACASS.EXE	RSMYsecq.XDB
FA-XEDIS.EXE	SCMDEF.XDB
FACASS.EXE	SCMIDid.DAT
FA-XEDIS.EXE SUB-TAB.EXE	SUB-TAB.XDB
RSMTDPS.EXE	SYLCHART.ASC
FACASS.EXE	TASKtigi.XDB
FACASS.EXE	TRAVTIME.DAT
DES-EDT3.EXE FACASS.EXE TRD-EDIT.EXE	TRDCOSTS.DAT
FACASS.EXE	TREEsecq.DAT

Table 3 (Cont'd)

Program	Data Files
DES-EDT3.EXE	TRWDti--.XDB
COST-DES.EXE UNC-FACT.EXE	UNCDSC.XDB
UNC-COMP.EXE UNC-FACT.EXE	UNC-FACT.XDB
FACASS.EXE UNC-FACT.EXE URRAPR.EXE	UNC-FACT.XDB
URRCOP.EXE URREDIT.EXE	URR.XDB
URRCOP.EXE	URRAPR.XDB
DES-EDT3.EXE	VALLIST.DAT
FA-XEDIS.EXE	WP-DESC.DAT

4 STANDARD SUBROUTINES

The following is an alphabetical list of all standard subroutines with detailed descriptions.

```
*************************************************************************
* Name       = ansist                                                   *
*                                                                       *
* Author     = Bobby Adams                                              *
*                                                                       *
* Function   = To perform various ansi-standard routines by sending the *
*              appropriate sequence of commands to the terminal.  Note, *
*              the command parameter (com) can be in upper or lower case.*
*                                                                       *
* Usage      = character com*3                                          *
*              integer*2 prm1, prm2                                     *
*              call ansist (com, prm1, prm2)                            *
*                                                                       *
* Parameters =                                                          *
*       com  - SGR (select graphic rendition)                           *
*       prm1 - 0 (turn all attributes off)                              *
*              1 (increase screen intensity)                            *
*              4 (dim screen intensity)                                 *
*              5 (blinking)                                             *
*              7 (reverse video)                                        *
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - HVP (horizontal and vertical position)                   *
*       prm1 - 1-24 (row in which to place cursor)                      *
*       prm2 - 1-80 (column in which to place cursor)                   *
*                                                                       *
*       com  - CUU (cursor up, doesn't change column position)          *
*       prm1 - number of rows to move up (won't move above top margin)  *
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - CUD (cursor down, doesn't change column position)        *
*       prm1 - number of rows to move down (won't go below bottom margin)*
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - CUF (cursor forward, doesn't change row position)        *
*       prm1 - number of cols to move right (won't go past right margin)*
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - CUB (cursor backward, doesn't change row position)       *
*       prm1 - number of cols to move left (won't go past left margin)  *
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - ED (erase in display, clear screen)                      *
*       prm1 - not used (use 0 for consistency)                         *
*       prm2 - not used (use 0 for consistency)                         *
*                                                                       *
*       com  - EL (erase in line)                                       *
*       prm1 - 0 (erase from cursor position to the end of line, inclusive)*
*              2 (erase the entire line)                                *
*       prm2 - line number to erase                                     *
*                                                                       *
* Returns    = None.                                                    *
*                                                                       *
* Calls      = fsc.lib:  upper  compac                                  *
*                                                                       *
* Commons    = None.                                                    *
*************************************************************************
```

```
*************************************************************************
* Name        = box (assembler routine)                                 *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Draws a box with an upper left corner at (x1,y1) and a lower *
*               right corner at (x2,y2).  The sides of the box are either *
*               single lines (type = 1) or double lines (type = 2).  A box can *
*               also be erased (type = 0).                              *
*                                                                       *
* Usage       = integer*2 x1, y1, x2, y2, type, att                     *
*               call box (x1, y1, x2, y2, type, att)                    *
*                                                                       *
* Parameters =                                                          *
*       x1    - row of the upper left hand corner                       *
*       y1    - column of the upper left hand corner                    *
*       x2    - row of the lower right hand corner (x2 must be > x1)    *
*       y2    - column of the lower right hand corner (y2 must be > y1) *
*       type  - 0 (blank lines, erase a box)                            *
*               1 (single line box)                                     *
*               2 (double line box)                                     *
*       att   - the screen attribute to use on the box (see wt)         *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = cd (convert date)                                       *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Converts a date from the internal storage format to the *
*               format specified.                                       *
*                                                                       *
* Usage       = character fmt*01, cd*11, date*11, newdate*11            *
*               newdate = cd (date, fmt)                                *
*                                                                       *
* Parameters =                                                          *
*       date  - the date to be converted (must be in the internal format of *
*               'YYYY/MMM/DD' i.e., '1985/001/01')                      *
*                                                                       *
*       fmt   - the date format to convert to (can be lower or upper case) *
*               'E' (English format of 'MMM/DD/YYYY' i.e., 'JAN/01/1985') *
*               'M' (metric format of 'DD/MMM/YYYY' i.e., '01/JAN/1985') *
*                                                                       *
* Returns     = The converted date stored in cd.  If the date passed is empty *
*               or in error, cd returns an empty date (i.e., '   /  /    ') *
*                                                                       *
* Calls       = fortran:  ichar                                         *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*****************************************************************************
* Name        = cdat                                                        *
*                                                                           *
* Author      = Kevin Stewart                                               *
*                                                                           *
* Function    = Converts a date from the internal database storage format to *
*               a real number (YYYY. + MM/12) that is to the nearest month. *
*                                                                           *
* Usage       = real*4    num, cdat                                         *
*               character date*11                                           *
*               num = cdat (date)                                           *
*                                                                           *
* Parameters =                                                              *
*       date - the date to be converted (in internal format 'YYYY/MMM/DD')  *
*                                                                           *
* Returns     = The date as a real number (YYYY. + MM/12).                  *
*                                                                           *
* Calls       = fsc.lib:  iconv                                             *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = chkbit (assembler)                                          *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Checks whether a bit is set or reset.                       *
*                                                                           *
* Usage       = integer*2 code, bit                                         *
*               logical*2 i, chkbit                                         *
*               i = chkbit (code, bit)                                      *
*                                                                           *
* Parameters =                                                              *
*       code - variable to check                                            *
*       bit  - the bit in variable (code) to check (0 - 15)                 *
*                                                                           *
* Returns     = True if the bit is set.                                     *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*****************************************************************************
* Name        = clrbox (clear box)                                          *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Clears the interior area of a box with the upper left-hand  *
*               corner at (x1,y1) and the lower right-hand corner at (x2,y2).*
*               The attribute of the interior is determined by attr.        *
*                                                                           *
* Usage       = integer*2 x1, y1, x2, y2, attr                              *
*               call clrbox (x1, y1, x2, y2, attr)                          *
*                                                                           *
* Parameters =                                                              *
*       x1.   - row of the upper left hand corner                           *
*       y1    - column of the upper left hand corner                        *
*       x2    - row of the lower right hand corner (x2 must be > x1)        *
*       y2    - column of the lower right hand corner (y2 must be > y1)     *
*       attr  - attribute of the interior of the box (see wt)               *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  scroll                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = clrmod (color mode - assembler)                             *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = To determine if the user's computer is in the color mode.   *
*                                                                           *
* Usage       = logical*2 clrmod, i                                         *
*               i = clrmod ()                                               *
*                                                                           *
* Parameters = None.                                                        *
*                                                                           *
* Returns     = clrmod                                                      *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*****************************************************************************
* Name        = cmp2sd (compare to system date)                             *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = To compare a date to the system date.                       *
*                                                                           *
* Usage       = character dat*11                                            *
*               integer*2 flag                                              *
*               call cmp2sd (dat, flag)                                     *
*                                                                           *
* Parameters =                                                              *
*       dat   - the date to be compared (must be in internal format i.e.,   *
*               'YYYY/MMM/DD')                                              *
*       flag  - indicates results of the comparison                         *
*               -1 (if the date is before the system date)                  *
*                0 (if the two dates are equal)                             *
*               +1 (if the date is after the system date)                   *
*                                                                           *
* Returns     = flag (indicates results of comparison).                     *
*                                                                           *
* Calls       = fsc.lib:  sysdat                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = command                                                     *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Execute a DOS command from within a program.                *
*                                                                           *
* Usage       = logical*2 error                                             *
*               character cmnd*(*)                                          *
*               call command (cmnd, error) or                               *
*               call command ('dir *.for'c,error)                           *
*                                                                           *
* Parameters =                                                              *
*    error    - true if an error occurred during execution.                 *
*    cmnd     - character variable containing the DOS command to execute (if*
*               a variable is used, the string contained by it must be      *
*               terminated with a null).                                    *
*    literal  - if the DOS command to be executed is contained in a literal *
*               string, the letter c must follow immediately after the string.*
*                                                                           *
* Returns     = if error is true, a message will have been written to the   *
*               file error.dat                                              *
*                                                                           *
* Calls       = fsc.lib:  rpterr                                            *
*               fortran:  system                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*************************************************************************
* Name        = compac                                                  *
*                                                                       *
* Author      = Bobby Adams                                             *
*                                                                       *
* Function    = Moves all blank characters in a string to the end of the*
*               string and returns the number of characters in the string.*
*                                                                       *
* Usage       = integer*2 num                                           *
*               character str*(*)                                       *
*               call compac (str, num)                                  *
*                                                                       *
* Parameters =                                                          *
*         str - the string variable of any length                       *
*         num - the number of non-blank characters in the string        *
*                                                                       *
* Returns     = It returns num, the length of character string stored in str.*
*                                                                       *
* Calls       = fortran:  len                                           *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = contain (logical function)                              *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Determines if a substring (sub) is contained within another*
*               string (str).                                           *
*                                                                       *
* Usage       = logical*2 contain, i                                    *
*               character sub*(*), str*(*)                              *
*               i = contain (str, sub)                                  *
*                                                                       *
* Parameters =                                                          *
*        str  - the string to search                                    *
*        sub  - the substring to search for                             *
*                                                                       *
* Returns     = contain is true if the substring is found within the target*
*               string.                                                 *
*                                                                       *
* Calls       = fortran:  len                                           *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = csort                                                   *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = This is an interface to a sort routine (written in c by Steve  *
*               Dorner and also contained in fsc.lib).  It will sort a  *
*               file using multiple keys in either ascending or descending  *
*               order.                                                  *
*                                                                       *
* Usage       = integer*4 addr1, addr2, locfar                          *
*               character control*(*), filen*(*)                        *
*                                                                       *
*               addr1 = locfar (control)                                *
*               addr2 = locfar (filen)                                  *
*               call csort (addr1, addr2)                               *
*                                                                       *
* Parameters =                                                          *
*    addr1    - the address of the control string                       *
*    addr2    - the address of the name of the file to be sorted        *
*    filen    - character string containing the name of the file to be sorted. *
*               the file name must be terminated with a null.           *
*    control  - a character variable containing the command string for the  *
*               sort.  the command string must be terminated with a null.  *
*               the format of control is as follows:                    *
*                  if first character is 's', second character specifies  *
*                  the field separator character.                         *
*                  followed by zero or more words of the form:            *
*                     [b][c|n][r][<m>][.<n>][-[<m2>][.<n2>]               *
*                     b ignore leading blanks                            *
*                     c ignore case                                      *
*                     n numeric comparison                               *
*                     r reverse order                                    *
*                     <m> field number                                   *
*                     .<n> begin column                                  *
*                     <m2> end field number                              *
*                     .<n2> end column                                   *
* Returns     = None.                                                   *
* Calls       = fsc.lib:  sort                                          *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = csrpos (assembler routine)                              *
*                                                                       *
* Author      = Dan Weidenfeld                                          *
*                                                                       *
* Function    = To position the cursor.                                 *
*                                                                       *
* Usage       = integer*2 row, col                                      *
*               call csrpos (row, col)                                  *
*                                                                       *
* Parameters =                                                          *
*         row - the row to position to - 1 (0-24)                       *
*         col - the column to position to - 1 (0-79)                    *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************

*************************************************************************
* Name        = cursor (assembler routine)                              *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Change the type of cursor used.                         *
*                                                                       *
* Usage       = integer*2 type                                          *
*               call cursor (type)                                      *
*                                                                       *
* Parameters =                                                          *
*        type - 0 (no cursor)                                           *
*               1 (underline cursor)                                    *
*               2 (dash cursor)                                         *
*               3 (overscore cursor)                                    *
*               4 (block cursor)                                        *
*               5 (bottom-half block cursor)                            *
*               6 (top-half block cursor)                               *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*****************************************************************************
* Name        = dantim (assembler routine)                                  *
*                                                                           *
* Author      = Dan Weidenfeld                                              *
*                                                                           *
* Function    = To get the system time.                                     *
*                                                                           *
* Usage       = integer*2 hours, min, secs                                  *
*               call gettim (hours, min, secs)                              *
*                                                                           *
* Parameters =                                                              *
*      hours - the hour of the day in military form (i.e., 1 pm is 13)      *
*      min   - the minutes                                                  *
*      secs  - the seconds                                                  *
*                                                                           *
* Returns     = the time of day.                                            *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = daydif (integer function)                                   *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Returns the number of days between two dates.               *
*                                                                           *
* Usage       = character sdate*11, edate*11                                *
*               integer*4 days, daydif                                      *
*               days = daydif (sdate, edate)                                *
*                                                                           *
* Parameters =                                                              *
*      sdate - the starting date (in the internal format)                   *
*      edate - the ending date (in the internal format)                     *
*                                                                           *
* Returns     = The number of days between the two dates.  If an invalid date *
*               is passed, the function returns the value -9,999.           *
*                                                                           *
* Calls       = fsc.lib:  iconv                                             *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*************************************************************************
* Name        = delay                                                   *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Delays program action for a specified number of seconds.*
*                                                                       *
* Usage       = integer*2 secs                                          *
*               call delay (secs)                                       *
*                                                                       *
* Parameters =                                                          *
*       secs - the number of seconds to delay                           *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fortran:  gettim                                        *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = dskcnf (assembler routine)                              *
*                                                                       *
* Author      = Paul Shih                                               *
*                                                                       *
* Function    = find out the disks type                                 *
*                                                                       *
* Usage       = integer*2 info(26)                                      *
*               call dskcnf(info)                                       *
*                                                                       *
* Returns     =                                                         *
*       Entry 1 is drive A                                              *
*             2 is drive B and so on .                                  *
*                                                                       *
* Commons     =                                                         *
*             INVALID_DSK          0                                    *
*             REMOTE_DSK          -1                                    *
*             LOCAL_RAM_DSK        1                                    *
*             LOCAL_FIXED_DSK      2                                    *
*             LOCAL_FLOPPY_DSK     3                                    *
*                                                                       *
*************************************************************************
```

```
*****************************************************************************
* Name        = erase                                                       *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Erases the lines between two rows (line1 and line2),        *
*               inclusive.                                                  *
* Usage       = integer*2 line1, line2, att                                 *
*               call erase (line1, line2, att)                              *
*                                                                           *
* Parameters =                                                              *
*      line1 - the erasing starts with this line (line1 must be <= line2)   *
*      line2 - the erasing ends with this line (line2 must be >= line1)     *
*      att   - the attribute to fill the erased field with                  *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  scroll                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = execute                                                     *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Suspends the program currently running and activates the    *
*               new program (filnam).  When the new program (filnam)        *
*               terminates, the original program is awakened and resumes its*
*               execution with the next instruction after the call to this  *
*               subroutine.                                                 *
*                                                                           *
* Usage       = logical*2 error                                             *
*               character filnam*(*)                                        *
*               call execute (filnam, error) or                             *
*               call execute ('prog2.exe'c, error)                          *
*                                                                           *
* Parameters =                                                              *
*    error    - true if an error occurred during execution                  *
*    filnam   - character variable containing the program name to execute (if *
*               a variable is used, the string contained by it must be      *
*               terminated with a null).                                    *
*    literal  - if the program name to be executed is contained in a literal *
*               string, the letter c must follow immediately after the string. *
*                                                                           *
* Returns     = if error is true, a message will have been written to the   *
*               file error.dat                                              *
*                                                                           *
* Calls       = fortran:  spawnlp                                           *
*               fsc.lib:  rpterr                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*************************************************************************
* Name        = fkeys                                                   *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Turns on/off the function display fields on line 25 (should *
*               be used in conjunction with fline).                     *
*                                                                       *
* Usage       = integer*2 f1, f2, f3, f4, f5, f6, f7, f8, f9, f10       *
*               call fkeys (f1, f2, f3, f4, f5, f6, f7, f8, f9, f10)    *
*                                                                       *
* Parameters =                                                          *
*   f1 - f10 - 0 (no change)                                            *
*              1 (turn display field off)                               *
*              2 (turn display field on) and displays (by default):     *
*                f1  - Help                                             *
*                f2  - Keys                                             *
*                f3  - Add                                              *
*                f4  - Delete                                           *
*                f5  - Edit                                             *
*                f6  - Find                                             *
*                f7  - List                                             *
*                f8  -                                                  *
*                f9  - Next                                             *
*                f10 - Exit                                             *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wt                                            *
*                                                                       *
* Commons     = ffldcom                                                 *
*                 fflds (array 10x2 of character*6).  if needed, the field to *
*                 be displayed can be changed by modifying the contents of *
*                 fflds (i.e., fflds(function key,2) = 'Change').       *
*************************************************************************


*************************************************************************
* Name        = fline                                                   *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Initially writes the function keys' display line.       *
*                                                                       *
* Usage       = call fline                                              *
*                                                                       *
* Parameters = None.                                                    *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wt                                            *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = frames (assembler routine)                              *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = To save up to five video screen windows into memory for later *
*               restoration.  When saving a window, the row and column  *
*               parameters specify the area to be saved.  When restoring a *
*               window, the row and column parameters specify the location the *
*               window should be restored to.                           *
*                                                                       *
* Usage       = integer*2 action, frame, r1, c1, r2, c2                 *
*               call frames (action, frame, r1, c1, r2, c2)             *
*                                                                       *
* Parameters =                                                          *
*     action - the action to take                                       *
*              1 (save the window to memory)                            *
*              anything else (restore the window from memory)           *
*     frame  - memory frame to save/restore the window to/from          *
*              1 - 5 (possible frames to use)                           *
*     r1     - row of the upper left hand corner of the window  (1-25)  *
*     c1     - column of the upper left hand corner of the window (1-80) *
*     r2     - row of the lower right hand corner of the window (1-25)  *
*     c2     - column of the lower right hand corner of the window (1-80) *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = gtinsk (get instruction key)                            *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Returns the ASCII integer value for the nonalphanumeric keys *
*               (e.g., function keys, Esc, Ins, etc.).  Alphanumeric keys are *
*               ignored.  Those keys that send two codes (i.e., a nul and then *
*               another number) will only return to the calling program the *
*               second number.                                          *
*                                                                       *
* Usage       = integer*2 key                                           *
*               call gtinsk (key)                                       *
*                                                                       *
* Parameters =                                                          *
*        key = ASCII integer value of the depressed key                 *
*                                                                       *
* Returns     = key (the ASCII integer value of the depressed nonalphanumeric *
*               key)                                                    *
*                                                                       *
* Calls       = fsc.lib:  inkey                                         *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = iconv (convert character string to integer)             *
*                                                                       *
* Author      = Bobby Adams                                             *
*                                                                       *
* Function    = To convert a character string of numbers to its integer *
*               equivalent.                                             *
*                                                                       *
* Usage       = character str*(*)                                       *
*               integer*4 num, iconv                                    *
*               logical*2 err                                           *
*               num = iconv (str, err)                                  *
*                                                                       *
* Parameters =                                                          *
*        str - the character string containing the number to convert    *
*        err - true if an error occurrs                                 *
*                                                                       *
* Returns     = iconv and err (if err, then iconv equals 0)             *
*                                                                       *
* Calls       = fortran:  ichar                                         *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = inkey (assembler routine)                               *
*                                                                       *
* Author      = Dan Weidenfeld                                          *
*                                                                       *
* Function    = To perform a single character read from the keyboard.   *
*                                                                       *
* Usage       = integer*2 i, inkey, echo                                *
*               i = inkey (echo)                                        *
*                                                                       *
* Parameters =                                                          *
*       echo - if equal to zero the character is not echoed to the screen.*
*                                                                       *
* Returns     = the ascii value of the character entered from the keyboard.*
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name       = inout (assembler routine)                                *
*                                                                       *
* Author     = Dan Weidenfeld                                           *
*                                                                       *
* Function   = To perform a single character write to the screen.       *
*                                                                       *
* Usage      = character char*01                                        *
*              call inout (char)                                        *
*                                                                       *
* Parameters =                                                          *
*       char - the character to be displayed on the screen              *
*                                                                       *
* Returns    = None.                                                    *
*                                                                       *
* Calls      = None.                                                    *
*                                                                       *
* Commons    = None.                                                    *
*************************************************************************


*************************************************************************
* Name       = isfile (assembler routine)                               *
*              integer*2 function isfile                                *
*                                                                       *
* Author     = Paul Shih                                                *
*                                                                       *
* Function   = To check file or subdirectory status                     *
*                                                                       *
* Usage      = integer*2 i                                              *
*              i = isfile('test.dat'c)                                  *
*                                                                       *
* Returns    = -1 : if file not found                                   *
*      Bit 0 on   : Read-Only file                                      *
*      Bit 1 on   : Hidden file                                         *
*      Bit 2 on   : System file                                         *
*      Bit 3 on   : Read-Only file                                      *
*      Bit 4 on   : Read-Only file                                      *
*                                                                       *
* Commons    = None.                                                    *
*************************************************************************
```

```
*************************************************************************
* Name       = line (assembler routine)                                 *
*                                                                       *
* Author     = Russ Hougland                                            *
*                                                                       *
* Function   = To draw a line (horizontal or vertical) directly into the *
*              video memory.  It will not draw a diagonal line.  This   *
*              routine allows the user to specify the beginning and ending *
*              characters of the line (in case it intercepts a box).    *
*                                                                       *
* Usage      = integer*2 r1, c1, r2, c2, begchr, endchr, type, attr     *
*              call line (r1, c1, r2, c2, begchr, endchr, type, attr)   *
*                                                                       *
* Parameters =                                                          *
*     r1     - starting row on the screen (1-25)                        *
*     c1     - starting column on the screen (1-80)                     *
*     r2     - ending row on the screen (1-25)                          *
*     c2     - ending column on the screen (1-80)                       *
*    begchr - the beginning character of the line (the decimal ascii value) *
*    endchr - the ending character of the line (the decimal ascii value) *
*     type   - the type of line to draw                                 *
*              1 - single line                                          *
*              2 - double line                                          *
*     attr   - attribute in which to display the line (see wt)          *
*                                                                       *
* Returns    = None.                                                    *
*                                                                       *
* Calls      = None.                                                    *
*                                                                       *
* Commons    = None.                                                    *
*************************************************************************
```

```
*************************************************************************
* Name        = pi (prompt for integer)                                 *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To write out a prompt and accept an integer response.  After  *
*               accepting the response, the prompt and reply are erased.  *
*               The routine places one space between the end of the prompt  *
*               and the start of the reply.                             *
*                                                                       *
* Usage       = integer*2 row, col, rlen, code, patt, ratt              *
*               integer*4 reply                                         *
*               character prompt*(*)                                    *
*               call pi (row, col, prompt, reply, rlen, code, patt, ratt)  *
*                                                                       *
* Parameters =                                                          *
*     row     - the row to display the prompt on                        *
*     col     - the column to begin displaying the prompt on            *
*               0 (center the prompt and reply)                         *
*     prompt  - the text to be displayed as a prompt                    *
*     reply   - the integer reply of the user                           *
*     rlen    - the maximum length of the user's reply                  *
*     code    - code can give special instructions for the reply entry. *
*               more than one can be passed by summing the options desired.  *
*               4 (user must enter a carriage return to enter a value)  *
*               8 (the number will not be written upon entry and exit from ri)  *
*               16 (the prompt will not be erased upon exit from pi)    *
*     patt    - the attribute to display the prompt in (see wt)         *
*     ratt    - the attribute to display the reply in (see wt)          *
*                                                                       *
* Returns     = reply                                                   *
*                                                                       *
* Calls       = fsc.lib:  chkbit ri  window  wt                         *
*               fortran:  len                                           *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*****************************************************************************
* Name        = pt (prompt for text)                                        *
*                                                                           *
* Author      = Kevin Stewart                                               *
*                                                                           *
* Function    = To write out a prompt and accept a text response.  After    *
*               accepting the response, the prompt and reply are erased.    *
*               The routine places one space between the end of the prompt  *
*               and the start of the reply.                                 *
*                                                                           *
* Usage       = integer*2 row, col, code, patt, ratt                        *
*               character prompt*(*), reply*(*)                             *
*               call pt (row, col, prompt, reply, code, patt, ratt)         *
*                                                                           *
* Parameters =                                                              *
*     row     - the row to display the prompt on                            *
*     col     - the column to begin displaying the prompt on                *
*               0 (center the prompt and reply)                             *
*     prompt  - the text to be displayed as a prompt                        *
*     reply   - the text reply of the user                                  *
*     code    - code can give special instructions for the reply entry.     *
*               more than one can be passed by summing the options desired. *
*               1 (the string is made upper case)                           *
*               2 (the string is packed -- no blanks)                       *
*               4 (user must enter a carriage return to enter a value)      *
*               8 (the text will not be written upon entry and exit from rt)*
*               16 (the prompt will not be erased upon exit from pi)        *
*     patt    - the attribute to display the prompt in (see wt)             *
*     ratt    - the attribute to display the reply in (see wt)              *
*                                                                           *
* Returns     = reply                                                       *
*                                                                           *
* Calls       = fsc.lib:  chkbit  rt  window  wt                            *
*               fortran:  len                                               *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*****************************************************************************
* Name       = rconv (convert character string to real)                     *
*                                                                           *
* Author     = Kevin Stewart                                                *
*                                                                           *
* Function   = To convert a character string of numbers to its real         *
*              equivalent.  The largest number that can be converted can    *
*              have up to 10 digits and 10 decimal places.                  *
*                                                                           *
* Usage      = character str*(*)                                            *
*              real*8    num, rconv                                       *
*              logical*2 err                                              *
*              num = rconv (str, err)                                       *
*                                                                           *
* Parameters =                                                              *
*       str - the character string containing the number to convert         *
*       err - true if an error occurrs                                      *
*                                                                           *
* Returns    = rconv and err (if err, then rconv equals 0)                  *
*                                                                           *
* Calls      = fsc.lib:  iconv  compac                                      *
*                                                                           *
* Commons    = None.                                                        *
*****************************************************************************
```

```
*************************************************************************
* Name        = rd (read date)                                          *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Reads a user's reply (date field) at a specified location *
*               (row,col).  Before the reply is read, the field is written *
*               in the attribute specified (attin); upon termination, the *
*               reply is rewritten in the attribute specified (attout).  *
*               The date is always returned in internal format          *
*               ('YYYY/0MM/DD').  This routine returns either a blank date *
*               ('0000/000/00') or a valid date, but never an invalid date. *
*                                                                       *
* Usage       = integer*2 row, col, code, attin, attout                 *
*               character fmt*01, date*11                               *
*               call rd (row, col, date, fmt, code, attin, attout)      *
*                                                                       *
* Parameters =                                                          *
*     row     - the row on which to read the date                       *
*     col     - the column on which to read the date                    *
*     date    - the date to be read                                     *
*     fmt     - the format of date for display and entry purposes (any case) *
*               'M' (metric DD/MMM/YYYY)                                *
*               'E' (English MMM/DD/YYYY)                               *
*     code    - code can give special instructions for the date entry.  *
*               more than one can be passed by summing the options desired. *
*               4 (user must enter a carriage return to enter a value)  *
*               8 (the date will not be written upon entry and exit from rd) *
*     attin   - the attribute to display the date in upon entry (see wt) *
*     attout  - the attribute to display the date in upon exit (see wt) *
*                                                                       *
* Returns     = code - indicates how rd was terminated (e.g., carriage return, *
*               up arrow, etc.)                                         *
*               date - the date entered by the user                     *
*                                                                       *
* Calls       = fsc.lib:  chkbit  window  ri  wi  rt  wt  wd            *
*               fortran:  ichar   mod                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = rf (read fixed real)                                    *
*                                                                       *
* Author      = Bobby Adams                                             *
*                                                                       *
* Function    = Reads a user's reply (real field) at a specified location *
*               (row,col).  Before the reply is read, the field can be  *
*               written in a specified attribute (attin); upon termination, *
*               the reply can be rewritten in a specified attribute (attout). *
*               Note the real number read is double precision.  Further, the *
*               total length of the number is len1 + len2 + 1 (decimal point). *
*                                                                       *
* Usage       = integer*2 row, col, len1, len2, code, attin, attout     *
*               real*8    num                                         *
*               call rf (row, col, num, len1, len2, code, attin, attout) *
*                                                                       *
* Parameters =                                                          *
*     row     - the row on which to read the number                     *
*     col     - the column on which to read the number                  *
*     num     - the number to be read                                   *
*     len1    - the maximum length of the left side of the number       *
*     len2    - the maximum length of the right side of the number      *
*     code    - code can give special instructions for the number entry. *
*               more than one can be passed by summing the options desired. *
*               4 (user must enter a carriage return to enter a value)  *
*               8 (the number will not be written upon entry and exit from rf) *
*     attin   - the attribute to display the number in upon entry (see wt) *
*     attout  - the attribute to display the number in upon exit (see wt) *
*                                                                       *
* Returns     = code - indicates how rf was terminated (e.g., carriage return, *
*               up arrow, etc.)                                         *
*               num  - the number entered by the user                   *
*                                                                       *
* Calls       = fsc.lib:  chkbit  window  cursor  csrpos  inkey  wt  wf *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = ri (read integer)                                       *
*                                                                       *
* Author      = Bobby Adams                                             *
*                                                                       *
* Function    = Reads a user's reply (integer field) at a specified location *
*               (row,col).  Before the reply is read, the field can be  *
*               written in a specified attribute (attin); upon termination, *
*               the reply can be rewritten in a specified attribute (attout). *
*                                                                       *
* Usage       = integer*2 row, col, len, code, attin, attout            *
*               integer*4 num                                           *
*               call ri(row, col, num, len, code, attin, attout)        *
*                                                                       *
* Parameters =                                                          *
*      row    - the row on which to read the number                     *
*      col    - the column on which to read the number                  *
*      num    - the number to be read                                   *
*      len    - the maximum length of the number                        *
*     code    - code can give special instructions for the number entry. *
*               more than one can be passed by summing the options desired. *
*               4 (user must enter a carriage return to enter a value)  *
*               8 (the number will not be written upon entry and exit from ri) *
*     attin   - the attribute to display the number in upon entry (see wt) *
*     attout  - the attribute to display the number in upon exit (see wt) *
*                                                                       *
* Returns     = code - indicates how ri was terminated (e.g., carriage return, *
*               up arrow, etc.)                                         *
*               num  - the number entered by the user                   *
*                                                                       *
* Calls       = fsc.lib:  chkbit  cursor  window  csrpos  inkey  wi  wt *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = rpterr (report error)                                   *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To append an error message to the report file (error.dat). *
*                                                                       *
* Usage       = character sn*(*), msg*(*)                               *
*               call rpterr (sn, msg)                                   *
*                                                                       *
* Parameters =                                                          *
*        sn   - subroutine name where error occurred                    *
*       msg   - error message to written to the error file              *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  sysdat  cd                                    *
*               fortran:  gettim  inquire  open  close                  *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*****************************************************************************
* Name        = rr (read real)                                              *
*                                                                           *
* Author      = Bobby Adams                                                 *
*                                                                           *
* Function    = Reads a user's reply (real field) at a specified location   *
*               (row,col).  Before the reply is read, the field can be      *
*               written in a specified attribute (attin); upon termination, *
*               the reply can be rewritten in a specified attribute (attout).*
*                                                                           *
* Usage       = integer*2 row, col, len, code, attin, attout                *
*               real*4    num                                             *
*               call rr(row, col, num, len, code, attin, attout)            *
*                                                                           *
* Parameters =                                                              *
*      row    - the row on which to read the number                         *
*      col    - the column on which to read the number                      *
*      num    - the number to be read                                       *
*      len    - the maximum length of the number                            *
*     code    - code can give special instructions for the number entry.    *
*               more than one can be passed by summing the options desired. *
*               4 (user must enter a carriage return to enter a value)      *
*               8 (the number will not be written upon entry and exit from rr) *
*     attin   - the attribute to display the number in upon entry (see wt)  *
*     attout  - the attribute to display the number in upon exit (see wt)   *
*                                                                           *
* Returns     = code - indicates how rr was terminated (e.g., carriage return, *
*               up arrow, etc.)                                             *
*               num  - the number entered by the user                       *
*                                                                           *
* Calls       = fsc.lib:  chkbit  window  cursor  csrpos  inkey  wt  wr     *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = rsscrn (assembler routine)                                  *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Restores the screen display previously saved by svscrn.     *
*                                                                           *
* Usage       = call rsscrn                                                 *
*                                                                           *
* Parameters = None                                                         *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*****************************************************************************
* Name       = rt (read text)                                               *
*                                                                           *
* Author     = Russ Hougland                                                *
*                                                                           *
* Function   = Reads a user's reply (text field) at a specified location    *
*              (row,col).  Before the reply is read, the field can be       *
*              written in a specified attribute (attin); upon termination,  *
*              the reply can be rewritten in a specified attribute (attout).*
*                                                                           *
* Usage      = integer*2 row, col, code, attin, attout                      *
*              character str*(*)                                            *
*              call rt (row, col, str, code, attin, attout)                 *
*                                                                           *
* Parameters =                                                              *
*      row   - the row on which to read the string                          *
*      col   - the column on which to read the string                       *
*      str   - the string to be read                                        *
*     code   - code can give special instructions for the text entry.       *
*              more than one can be passed by summing the options desired.  *
*              1 (the string is made upper case)                            *
*              2 (the string is packed -- no blanks)                        *
*              4 (user must enter a carriage return to enter a value)       *
*              8 (the number will not be written upon entry and exit from rt)*
*     attin  - the attribute to display the string in upon entry (see wt)   *
*     attout - the attribute to display the string in upon exit (see wt)    *
*                                                                           *
* Returns    = code - indicates how rt was terminated (e.g., carriage return,*
*              up arrow, etc.)                                              *
*              str  - the string entered by the user                        *
*                                                                           *
* Calls      = fsc.lib:  chkbit  cursor  csrpos  compac  inkey  wt          *
*              fortran:  ichar                                              *
*                                                                           *
* Commons    = None.                                                        *
*****************************************************************************


*****************************************************************************
* Name       = rv (assembler routine)                                       *
*                                                                           *
* Author     = Russ Hougland                                                *
*                                                                           *
* Function   = To convert an attribute to reverse video.  It reverses the   *
*              foreground and background RGB bits while leaving the intensity*
*              and blinking bits unchanged.                                 *
*                                                                           *
* Usage      = integer*2 newatt, oldatt, rv                                 *
*              newatt = rv (oldatt)                                         *
*                                                                           *
* Parameters =                                                              *
*     oldatt - the old attribute that needs reversing                       *
*                                                                           *
* Returns    = rv (the new attribute after reversing).                      *
*                                                                           *
* Calls      = None.                                                        *
*                                                                           *
* Commons    = None.                                                        *
*****************************************************************************
```

```
*****************************************************************************
* Name        = scroll (assembler routine)                                  *
*                                                                           *
* Author      = Dan Weidenfeld                                              *
*                                                                           *
* Function    = To scroll a window on the screen.                           *
*                                                                           *
* Usage       = integer*2 r1, c1, r2, c2, action, att                       *
*               call scroll (r1, c1, r2, c2, action, att)                   *
*                                                                           *
* Parameters =                                                              *
*     r1      - row of the upper left hand corner of the window to be       *
*               scrolled - 1 (0-24)                                         *
*     c1      - column of the upper left hand corner of the window to be    *
*               scrolled - 1 (0-79)                                         *
*     r2      - row of the lower right hand corner of the window to be      *
*               scrolled - 1 (0-24)                                         *
*     c2      - column of the lower right hand corner of the window to be   *
*               scrolled - 1 (0-79)                                         *
*     action  - the action to perform                                       *
*               +n (scroll up n lines)                                      *
*               -n (scroll down n lines)                                    *
*                0 (scroll the entire window)                               *
*     att     - the attribute of blank lines added after scrolling (see wt) *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = setatt                                                      *
*                                                                           *
* Author      = Bobby Adams                                                 *
*                                                                           *
* Function    = To initialize the values of the attribute variables used by *
*               the screen read and write routines.  This routine must be   *
*               called once at the beginning of any program that uses the   *
*               attribute variables stored in attrib.inc.  The color values of *
*               the variables can be changed by changing the file, color.dat. *
*                                                                           *
* Usage       = call setatt                                                 *
*                                                                           *
* Parameters = None.                                                        *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  clrmod                                            *
*                                                                           *
* Commons     = attrib (attrib.inc) - contains the values to initialize     *
*****************************************************************************
```

```
*****************************  ***************************************************
* Name        = slen (integer function)                                          *
*                                                                                *
* Author      = Kevin Stewart                                                    *
*                                                                                *
* Function    = Find the length of a character string within a character         *
*               variable (i.e., the position of the last nonblank character).    *
*                                                                                *
* Usage       = integer*2 i, slen                                                *
*               character str*(*)                                                *
*               i = slen (str)                                                   *
*                                                                                *
* Parameters =                                                                   *
*         str - character variable containing the string being passed            *
*                                                                                *
* Returns     = slen - the length of the character string (0 if all blanks)      *
*                                                                                *
* Calls       = None.                                                            *
*                                                                                *
* Commons     = None.                                                            *
**********************************************************************************


* ***********************************************************************************
* Name        = svscrn (assembler routine)                                       *
*                                                                                *
* Author      = Russ Hougland                                                    *
*                                                                                *
* Function    = Saves the screen display in a buffer area for later              *
*               restoration by rsscrn.                                           *
*                                                                                *
* Usage       = call svscrn                                                      *
*                                                                                *
* Parameters = None                                                              *
*                                                                                *
* Returns     = None.                                                            *
*                                                                                *
* Calls       = None.                                                            *
*                                                                                *
* Commons     = None.                                                            *
**********************************************************************************


**********************************************************************************
* Name        = sysdat (system date)                                             *
*                                                                                *
* Author      = Kevin Stewart                                                    *
*                                                                                *
* Function    = To return the system date in the internal storage format.        *
*                                                                                *
* Usage       = character*11 dat                                                 *
*               call sysdat (dat)                                                *
*                                                                                *
* Parameters =                                                                   *
*         dat - the system date in internal format ('YYYY/MMM/DD')               *
*                                                                                *
* Returns     = the system date (dat).                                           *
*                                                                                *
* Calls       = fortran:  getdat                                                 *
*                                                                                *
* Commons     = None.                                                            *
**********************************************************************************
```

```
*************************************************************************
* Name        = today (assembler routine)                               *
*                                                                       *
* Author      = Dan Weidenfeld                                          *
*                                                                       *
* Function    = To get the system date.                                 *
*                                                                       *
* Usage       = integer*2 year, month, day                              *
*               call today (year, month, day)                           *
*                                                                       *
* Parameters =                                                          *
*      year   - the year of the system date (e.g., 1986)                *
*                                                                       *
*      month  - the month of the system date (e.g., 12)                 *
*                                                                       *
*      day    - the day of the system date (e.g., 25)                   *
*                                                                       *
*                                                                       *
* Returns     = the system date (year,month,day).                       *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = upper                                                   *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Converts to upper case all letters in the parameter.    *
*                                                                       *
* Usage       = character*(*) str                                       *
*               call upper (str)                                        *
*                                                                       *
* Parameters =                                                          *
*        str - the character string to be converted to upper case       *
*                                                                       *
* Returns     = str (all letters in upper case)                         *
*                                                                       *
* Calls       = fortran:  ichar  char                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*****************************************************************************
* Name        = wd (write date)                                             *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Write a date at a specified location (row,col) in the form  *
*               specified.  The date should be passed in the internal format*
*               ('YYYY/MMM/DD').                                            *
*                                                                           *
* Usage       = integer*2 row, col, attr                                    *
*               character fmt*01, date*11                                   *
*               call wd (row, col, date, fmt, attr)                         *
*                                                                           *
* Parameters =                                                              *
*      row    - the row on which to write the date                          *
*      col    - the column on which to write the date                       *
*               0 (to center the date on the line)                          *
*      date   - the date to be written                                      *
*      fmt    - the format of date (can be lower or upper case)             *
*               'M' (metric DD/MMM/YYYY)                                    *
*               'E' (English MMM/DD/YYYY)                                   *
*      attr   - the attribute to display the date in (see wt)               *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  cd  wt                                            *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = wf (write fixed real)                                       *
*                                                                           *
* Author      = Bobby Adams                                                 *
*                                                                           *
* Function    = Write a real at a specified location (row,col) using the    *
*               desired attribute.  Note the number written is double       *
*               precision.  Further, the total length of the number is      *
*               len1 + len2 + 1 (decimal point).                            *
*                                                                           *
* Usage       = integer*2 row, col, attr, len1, len2                        *
*               real*8    num                                             *
*               call wf (row, col, num, len1, len2, attr)                   *
*                                                                           *
* Parameters =                                                              *
*      row    - the row on which to write the number                        *
*      col    - the column on which to write the number                     *
*               0 (to center the number on the line)                        *
*      num    - the number to be written                                    *
*      len1   - the length of the left side of the number in                *
*      len2   - the length of the right side of the number in               *
*      attr   - the attribute to display the date in (see wt)               *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  compac  wt                                        *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*****************************************************************************
* Name        = wi (write integer)                                          *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Write an integer at a specified location (row,col) using the *
*               desired attribute.                                          *
*                                                                           *
* Usage       = integer*2 row, col, attr, len                               *
*               integer*4 num                                               *
*               call wi (row, col, num, len, attr)                          *
*                                                                           *
* Parameters =                                                              *
*      row    - the row on which to write the number                        *
*      col    - the column on which to write the number                     *
*               0 (to center the number on the line)                        *
*      num    - the number to be written                                    *
*      len    - the length of the field to display the number in            *
*      attr   - the attribute to display the number in (see wt)             *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = fsc.lib:  wi                                                *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************


*****************************************************************************
* Name        = window (assembler routine)                                  *
*                                                                           *
* Author      = Russ Hougland                                               *
*                                                                           *
* Function    = Set the video attribute for a window directly into video    *
*               memory.                                                     *
*                                                                           *
* Usage       = integer*2 r1, c1, r2, c2, attr                              *
*               call window (r1, c1, r2, c2, attr)                          *
*                                                                           *
* Parameters =                                                              *
*       r1    - row of the upper left hand corner                           *
*       c1    - column of the upper left hand corner                        *
*       r2    - row of the lower right hand corner (x2 must be > x1)        *
*       c2    - column of the lower right hand corner (y2 must be > y1)     *
*       attr  - the attribute of the window (see wt)                        *
*                                                                           *
* Returns     = None.                                                       *
*                                                                           *
* Calls       = None.                                                       *
*                                                                           *
* Commons     = None.                                                       *
*****************************************************************************
```

```
*******************************************************************************
* Name        = wr (write real)                                               *
*                                                                             *
* Author      = Bobby Adams                                                   *
*                                                                             *
* Function    = Write a real at a specified location (row,col) using the      *
*               desired attribute.                                            *
*                                                                             *
* Usage       = integer*2 row, col, attr, len                                 *
*               real*4    num                                                 *
*               call wr (row, col, num, len, attr)                            *
*                                                                             *
* Parameters =                                                                *
*      row    - the row on which to write the number                          *
*      col    - the column on which to write the number                       *
*               0 (to center the number on the line)                          *
*      num    - the number to be written                                      *
*      len    - the length of the field to display the number in              *
*      attr   - the attribute to display the date in (see wt)                 *
*                                                                             *
* Returns     = None.                                                         *
*                                                                             *
* Calls       = fsc.lib:  wt                                                  *
*                                                                             *
* Commons     = None.                                                         *
*******************************************************************************


*******************************************************************************
* Name        = wsd (write screen date)                                       *
*                                                                             *
* Author      = Kevin Stewart                                                 *
*                                                                             *
* Function    = To write a prompt and a variable to the screen.               *
*                                                                             *
* Usage       = integer*2 row, col, patt, vatt                                *
*               character prompt*(*), dat*11, fmt*01                          *
*               call wsd (row, col, prompt, dat, fmt, patt, vatt)             *
*                                                                             *
* Parameters =                                                                *
*     row     - the row position of the date                                  *
*     col     - the column position of the date                               *
*               0 (center the prompt and the date in the row)                 *
*     prompt  - the prompt (displayed at row,col-plen-1)                      *
*     dat     - the date variable to display                                  *
*     fmt     - the format to display the date in (English or metric)         *
*     patt    - the attribute to use when displaying the prompt               *
*     vatt    - the attribute to use when displaying the date                 *
*                                                                             *
* Returns     = None.                                                         *
*                                                                             *
* Calls       = fsc.lib:  wd  wt                                              *
*                                                                             *
* Commons     = None.                                                         *
*******************************************************************************
```

```
*************************************************************************
* Name        = wsf (write screen fixed real)                           *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To write a prompt and a variable to the screen.         *
*                                                                       *
* Usage       = integer*2 row, col, len1, len2, patt, vatt              *
*               character prompt*(*)                                    *
*               real*8    var                                           *
*               call wsf (row, col, prompt, var, len1, len2, patt, vatt)*
*                                                                       *
* Parameters =                                                          *
*     row     - the row position of the variable                        *
*     col     - the column position of the variable                     *
*               0 (center the prompt and the variable in the row)       *
*     prompt  - the prompt (displayed at row,col-plen-1)                *
*     var     - the variable to display                                 *
*     len1    - the maximum length of the left side of the number       *
*     len2    - the maximum length of the right side of the number      *
*     patt    - the attribute to use when displaying the prompt         *
*     vatt    - the attribute to use when displaying the variable       *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wf  wt                                        *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = wsi (write screen integer)                              *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To write a prompt and a variable to the screen.         *
*                                                                       *
* Usage       = integer*2 row, col, vlen, patt, vatt                    *
*               character prompt*(*)                                    *
*               integer*4 var                                           *
*               call wsi (row, col, prompt, var, vlen, patt, vatt)      *
*                                                                       *
* Parameters =                                                          *
*     row     - the row position of the variable                        *
*     col     - the column position of the variable                     *
*               0 (center the prompt and the variable in the row)       *
*     prompt  - the prompt (displayed at row,col-plen-1)                *
*     var     - the variable to display                                 *
*     vlen    - the length of the variable                              *
*     patt    - the attribute to use when displaying the prompt         *
*     vatt    - the attribute to use when displaying the variable       *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wi  wt                                        *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = wsr (write screen real)                                 *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To write a prompt and a variable to the screen.         *
*                                                                       *
* Usage       = integer*2 row, col, vlen, patt, vatt                    *
*               character prompt*(*)                                    *
*               real*4    var                                           *
*               call wsr (row, col, prompt, var, vlen, patt, vatt)      *
*                                                                       *
* Parameters =                                                          *
*     row     - the row position of the variable                        *
*     col     - the column position of the variable                     *
*               0 (center the prompt and the variable in the row)       *
*     prompt  - the prompt (displayed at row,col-plen-1)                *
*     var     - the variable to display                                 *
*     vlen    - the length of the variable                              *
*     patt    - the attribute to use when displaying the prompt         *
*     vatt    - the attribute to use when displaying the variable       *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wr  wt                                        *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************


*************************************************************************
* Name        = wst (write screen text)                                 *
*                                                                       *
* Author      = Kevin Stewart                                           *
*                                                                       *
* Function    = To write a prompt and a variable to the screen.         *
*                                                                       *
* Usage       = integer*2 row, col, patt, vatt                          *
*               character prompt*(*), var*(*)                           *
*               call wst (row, col, prompt, var, patt, vatt)            *
*                                                                       *
* Parameters =                                                          *
*     row     - the row position of the variable                        *
*     col     - the column position of the variable                     *
*               0 (center the prompt and the variable in the row)       *
*     prompt  - the prompt (displayed at row,col-plen-1)                *
*     var     - the variable to display                                 *
*     patt    - the attribute to use when displaying the prompt         *
*     vatt    - the attribute to use when displaying the variable       *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = fsc.lib:  wt                                            *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
* Name        = wt (write text -- assembler routine)                    *
*                                                                       *
* Author      = Russ Hougland                                           *
*                                                                       *
* Function    = Write a string at a specified location (row,col) using the *
*               desired attribute.                                      *
*                                                                       *
* Usage       = integer*2 row, col, attr, len                           *
*               character str*(*)                                       *
*               call wt (row, col, str, len, attr)                      *
*                                                                       *
* Parameters =                                                          *
*      row    - the row on which to write the string                    *
*      col    - the column on which to write the string                 *
*               0  (center the field including trailing blanks)         *
*      str    - the string to be written                                *
*      len    - the length of the field to display the string in        *
*      attr   - the attribute to display the string in                  *
*               to determine the number for a color use the following:  *
*                  foreground color:  blue    1                       *
*                                     green   2                         *
*                                     red     4                         *
*                  intensity       :          8                       *
*                  background color:  blue   16                       *
*                                     green  32                         *
*                                     red    64                         *
*                  blinking        :        128                       *
*                                                                       *
*               to get the desired color, simply sum the numbers of the desired*
*               attributes, for example: to get a blinking white text on a *
*               black background attr = 1 + 2 + 4 (white foreground)    *
*                                     + 0         (black background)    *
*                                     + 128       (blinking)            *
*                                                                       *
* Returns     = None.                                                   *
*                                                                       *
* Calls       = None.                                                   *
*                                                                       *
* Commons     = None.                                                   *
*************************************************************************
```

```
*************************************************************************
*BTRIEVE CALLS                                                          *
*                                                                       *
*FUNCTION   :  INTERFACE WITH THE BTRIEVE TO ACCESS A SPECIFIED FILE    *
*                                                                       *
*USAGE      :  INTEGER*2 STS,FILBLK(64),REC(N),LREC,LKEY,OP,KEYNUM      *
*              CHARACTER KEY*N                                          *
*              CALL BTREE(FILBLK,OP,KEYNUM,KEY,LKEY,REC,LREC,STS)       *
*                                                                       *
*PARAMETERS:   FILBLK - STORES THE FILE NAME AND PATH, CURRENT RECORD   *
*                       POINTER                                         *
*                                                                       *
*              OP     - THE TYPE OF OPERATION TO BE EXECUTED           *
*                       (i.e., OPEN,UPDATE, etc.)                       *
*                                                                       *
*              KEYNUM - THE KEY PATH USED IN ALL FILE OPERATIONS       *
*                                  ** NOTE **                           *
*                       THE KEYNUM USED MUST BE A VALID KEY SPECIFED    *
*                       DURING THE FILES INITIAL OPERATION.             *
*                                                                       *
*              KEY    - CONTAINS THE KEY FIELD OF THE RECORD           *
*                                                                       *
*              LKEY   - LENGTH OF KEY FIELD IN BYTES                   *
*                                                                       *
*              REC    - INTEGER*2 ARRAY CONTAINING RECORD DATA         *
*                                                                       *
*              LREC   - LENGTH OF RECORD IN BYTES                      *
*                                                                       *
*              STS    - CONTAINS THE STATUS OF THE OPERATION PERFORMED *
*                       RETURN ZERO IF NORMAL OPERATION                 *
*                                                                       *
*NOTE       :  WHEN CREATING AND OPENING FILES, THE FILE NAME IS IN THE  *
*              'KEY' FIELD AND THE FILE NAME LENGTH IS IN THE 'LKEY'    *
*              FILED, example:                                          *
*                                                                       *
*              CALL BTREE (FILBLK,0,0,'FILE.XDB',8,REC,LREC,STS)        *
*************************************************************************


*************************************************************************
* BTRIEVE OPERATIONS                                                    *
*                                                                       *
* (0)       - OPEN                                                      *
* PURPOSE   - TO OPEN A BTRIEVE FILE                                    *
*                                                                       *
* (1)       - CLOSE                                                     *
* PURPOSE   - TO CLOSE A BTRIEVE FILE                                   *
*                                                                       *
* (2)       - INSERT                                                    *
* PURPOSE   - TO INSERT A RECORD IN A FILE                              *
*                                                                       *
* (3)       - UPDATE                                                    *
* PURPOSE   - TO UPDATE AN EXISTING RECORD IN A BTRIEVE FILE            *
*                                                                       *
* (4)       - DELETE                                                    *
* PURPOSE   - TO DELETE AN EXISTING RECORD IN A BTRIEVE FILE            *
*                                                                       *
* (5)       - GET EQUAL                                                 *
* PURPOSE   - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING    *
*               TO A SPECIFIC KEY VALUE                                 *
*                                                                       *
* (6)       - GET NEXT                                                  *
```

```
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE THAT FOLLOWS          *
*            THE "CURRENT RECORD"                                          *
*                                                                          *
* (7)      - GET PREVIOUS                                                  *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE THAT PRECEDES        *
*            THE "CURRENT RECORD"                                          *
*                                                                          *
* (8)      - GET GREATER                                                   *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE KEY VALUE THAT IS GREATER THAN A SPECIFIC KEY VALUE    *
*                                                                          *
* (9)      - GET GREATER OR EQUAL                                          *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE KEY VALUE THAT IS GREATER OR EQUAL TO A SPECIFIC       *
*            KEY VALUE                                                     *
*                                                                          *
* (10)     - GET LESS THAN                                                 *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE KEY VALUE WHICH IS LESS THAN A SPECIFIC KEY VALUE      *
*                                                                          *
* (11)     - GET LESS THAN OR EQUAL                                        *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE KEY VALUE WHICH IS LESS THAN OR EQUAL TO A SPECIFIC    *
*            KEY VALUE                                                     *
*                                                                          *
* (12)     - GET LOWEST                                                    *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE LOWEST KEY VALUE FOR A SPECIFIED ACCESS PATH.          *
*                                                                          *
* (13)     - GET HIGHEST                                                   *
* PURPOSE  - TO RETRIEVE A RECORD FROM A BTRIEVE FILE CORRESPONDING        *
*            TO THE HIGHEST KEY VALUE FOR A SPECIFIED ACCESS PATH.         *
*                                                                          *
* (14)     - CREATE                                                        *
* PURPOSE  - TO CREATE A BTRIEVE FILE WITH THE SPECIFIED SET OF            *
*            CHARACTERISTICS                                               *
*                                                                          *
* (15)     - STAT                                                          *
* PURPOSE  - TO RETRIEVE A SPECIFIED FILE'S CHARACTERISTICS                *
*                                                                          *
* (16)     - EXTEND                                                        *
* PURPOSE  - TO EXTEND A BTRIEVE FILE TO A SECOND LOGICAL DISK DRIVE       *
*                                                                          *
* (17)     - SET DIRECTORY                                                 *
* PURPOSE  - TO SET THE CURRENT DIRECTORY TO A SPECIFIED VALUE             *
*                                                                          *
* (18)     - GET DIRECTORY                                                 *
* PURPOSE  - TO RETRIEVE THE "CURRENT" DIRECTORY                           *
*                                                                          *
* (19)     - BEGIN TRANSACTION                                             *
* PURPOSE  - TO MARK THE BEGINNING OF A SET OF LOGICALLY RELATED BTRIEVE   *
*            OPERATIONS.                                                   *
*                                                                          *
* (20)     - END TRANSACTION                                               *
* PURPOSE  - TO COMPLETE A TRANSACTION AND COMMIT THE OPERATIONS PERFORMED  *
*            SINCE THE TRANSACTION BEGAN                                   *
*                                                                          *
* (21)     - ABORT TRANSACTION                                             *
* PURPOSE  - TO REMOVE ALL OPERATIONS PERFORMED SINCE BEGINNING OF         *
*            AN ACTIVE TRANSACTION, AND TO TERMINATE THE TRANSACTION.      *
*                                                                          *
* (22)     - GET POSITION                                                  *
* PURPOSE  - TO RETURN THE PHYSICAL POSITION OF THE RECORD IN THE BTRIEVE   *
```

```
*            FILE THAT HAS BEEN ESTABLISHED AS THE "CURRENT RECORD"          *
*                                                                            *
* (23)      - GET DIRECT                                                     *
* PURPOSE   - TO RETRIEVE THE DATA RECORD POSITIONED AT A SPECIFED           *
*             ADDRESS IN THE BTRIEVE FILE.                                   *
*                                                                            *
* (24)      - STEP DIRECT                                                    *
* PURPOSE   - TO RETRIEVE THE DATA RECORD IN THE LOCATION PHYSICALLY         *
*             FOLLOWING THE CURRENT RECORD IN THE BTRIEVE FILE.              *
*                                                                            *
* (25)      - STOP                                                           *
* PURPOSE   - TO TERMINATE THE RECORD MANAGER AND REMOVE IT FROM MEMORY.     *
******************************************************************************


******************************************************************************
* BTRIEVE ERROR CODES                                                        *
*                                                                            *
* (For detailed information see Btrieve menu Appendix B ERROR CONDITIONS)    *
*                                                                            *
* A utility program (STS.EXE) is available to prompt Btrieve Error           *
* Condition and Fortran Runtime Error.                                       *
*                                                                            *
* 01      INVALID OPERATION                                                  *
* 02      I/O ERROR                                                          *
* 03      NO OPEN                                                            *
* 04      KEY NOT FOUND                                                      *
* 05      DUPLICATES ERROR                                                   *
* 06      INVALID KEY NUMBER                                                 *
* 07      DIFFERENT KEY NUMBER                                               *
* 08      INVALID POSITIONING                                                *
* 09      END OF FILE                                                        *
* 10      MODIFIABLE ERROR                                                   *
* 11      INVALID FILE NAME                                                  *
* 12      FILE NOT FOUND                                                     *
* 13      EXTENSION ERROR                                                    *
* 14      PRE-OPEN ERROR                                                     *
* 15      PRE-IMAGE ERROR                                                    *
* 16      EXPANSION ERROR                                                    *
* 17      CLOSE ERROR                                                        *
* 18      DISK FULL                                                          *
* 19      UNRECOVERABLE ERROR                                                *
* 20      RECORD MANAGER INACTIVE                                            *
* 21      KEY BUFFER ERROR                                                   *
* 22      RECORD BUFFER (DATA BUFFER NOT LONG ENOUGH)                        *
* 23      POSITION BLOCK (MUST BE 128 BYTES)                                 *
* 24      PAGE SIZE (MUST BE MULTIPLE OF 512)                                *
* 25      CREATE I/O ERROR                                                   *
* 26      NUMBER OF KEYS (PAGE SIZE AND NUM OF KEYS DO NOT MATCH)            *
* 27      KEY POSITION                                                       *
* 28      RECORD LENGTH (NO GREATER THAN PAGE SIZE - 6)                      *
* 29      KEY LENGTH (1-255)                                                 *
* 30      BTRIEVE FILE NAME (INVALID BTRIEVE FILE)                           *
* 31      EXTEND ERROR (ALREADY EXTENDED)                                    *
* 32      EXTEND I/O ERROR                                                   *
* 33      EXTEND DRIVE ERROR                                                 *
* 34      EXTEND NAME                                                        *
* 35      DIRECTORY ERROR                                                    *
* 36      TRANSACTON ERROR (/T OPTION NOT SPECIFIED WHEN REC-MAN LOADED)     *
* 37      BEGIN TRANSACTION (TRANSACTION ALREADY ACTIVE)                     *
* 38      TRANSACTION CONTROL FILE                                           *
* 39      END/ABORT ERROR                                                    *
* 40      TRANSACTION MAX FILES (UP TO 8 FILES MAY BE UPDATED DURING TRANS)  *
```

```
* 41      TRANSACTION OPEN/CLOSE                                              *
* 42      INCOMPLETE ACCELERATED ACCESS                                       *
* 43      INVALID DATA RECORD ADDRESS                                         *
* 44      NULL KEY PATH                                                       *
* 45      INCONSISTENT KEY FLAGS                                              *
  46      ACCESS DENIED                                                       *
* 47      MAXIMUM OPEN FILES                                                  *
* 48      INVALID ALTERNATE SEQUENCE DEFINITION                               *
* 49      KEY TYPE ERROR                                                      *
* 50      OWNER ALREADY SET                                                   *
* 51      INVALID OWNER                                                       *
* 52      ERROR WRITING CACHE                                                 *
* 53      INVALID INTERFACE                                                   *
* 54      VARIABLE PAGE UNREADABLE                                            *
* 80      CONFLICT                                                            *
* 81      LOCK FULL                                                           *
* 82      LOST POSITION                                                       *
* 83      READ OUTSIDE TRANSACTION                                            *
* 84      RECORD IN USE                                                       *
* 85      FILE IN USE                                                         *
* 86      FILE FULL                                                           *
* 87      HANDLE FULL                                                         *
* 88      MODE ERROR                                                          *
* 89      NAME ERROR                                                          *
* 90      DEVICE FULL                                                         *
* 91      SERVER ERROR                                                        *
* 92      TRANSACTION FULL                                                    *
* 99      DEMO ERROR                                                          *
*                                                                             *
*******************************************************************************
```

5 STANDARD COMMON BLOCKS

The following is an alphabetical list of the standard common blocks and a short description of each block.

```
FILE NAME: AMSCOD.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: AMSCOD.INC
NUMBER OF KEYS:  2
RECORD SIZE : 52
PAGE SIZE   : 512
RECORD NAME : AMSREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     AMSKEY      1         2     N   N    S
 1     AMSCOD      3         10    N   Y    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
AMSKEY          S   2          1       AMS ID
AMSCOD          S   10         3       AMS CODE
AMSDSC          S   40         13      AMS DESCRIPTION
```

```
FILE NAME: AMSDSC.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: AMSDSC.INC
NUMBER OF KEYS:  2
RECORD SIZE : 50
PAGE SIZE   : 512
RECORD NAME : AMSRC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     AMSID       1         10    N   Y    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
AMSID           S   10         1       AMS CODE
AMSDSC          S   40         11      AMS DESCRIPTION
```

```
FILE NAME: AMSF4C.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: AMSF4C.INC
NUMBER OF KEYS:  2
RECORD SIZE : 18
PAGE SIZE   : 512
RECORD NAME : F4CREC

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     F4CCOD       1         7    N   N    S
 1     AMSEQV       9        10    Y   Y    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
F4CCOD          S    7          1       F4C CODE
AMSEQV          S   10          9       AMS NUMBER
```

```
FILE NAME: APRCOD.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: APRCOD.INC
NUMBER OF KEYS:  2
RECORD SIZE : 52
PAGE SIZE   : 512
RECORD NAME : APRREC

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     APRKEY       1         2    N   Y    S
 1     APRCOD       3        10    N   Y    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
APRKEY          S    2          1       APPROPRIATION ID
APRCOD          S   10          3       APP. CODE
APRDSC          S   40         13       APP. DESCRIPTION
```

```
FILE NAME: AREA_TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 34
PAGE SIZE   : 512
RECORD NAME : ARDAT

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     CODE         1        2     N   Y    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
CODE            S    2          1       AREA ID
DEF             S   30          3       AREA DESCRIPTION
```

```
FILE NAME: BT-PASS.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 1

**** FILE INFORMATION ****
 LINE      FORMAT      FIELDS
*****   **********  **********
  1         A4         TTGG

***** FIELD INFORMATION *****
FIELD          TYPE LENGTH  DESCRIPTION
NAME                (BYTES)
*****          **** ******* ************
TTGG            S    4       BASIC TASK SUMMARY FILE NAME SPECIFIER

**** NOTES ****
THIS VARIABLE REPLACES POSITIONS 5 to 8 IN THE FILE NAME 'BTSMttgg.XDB'
```

```
FILE NAME: CLASLST.DAT
FILE TYPE: TEXT
NUMBER OF KEYS:  1
NUMBER OF LINES: 1

**** FILE INFORMATION ****
 LINE      FORMAT      FIELDS
*****   **********  **********
  1       I2,A30       INDEX,DESCR

***** FIELD INFORMATION *****
FIELD          TYPE LENGTH  DESCRIPTION
NAME                (BYTES)
*****          **** ******* ************
INDEX           S     2     CLASSIFICATION ID ** KEY **
DESCR           S    30     CLASSIFICATION DESCRIPTION
```

```
FILE NAME: DES-BTSM.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 34
PAGE SIZE   : 512
RECORD NAME : DESDAT

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     KEY         1        4     Y   N    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
TREEID          S    2          1      TREE TABLE ID
GRPID           S    2          3      BASIC TASK TABLE SUMMARY ID
DESC            S   30          5      BASIC TASK SUMMARY TABLE NAME

**** NOTES ****
THE FIELDS 'TREEID' AND 'GRPID' ARE COMBINED TO FORM THE KEY FIELD
```

```
FILE NAME: DES-TASK.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 34
PAGE SIZE   : 512
RECORD NAME : DESDAT

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     KEY         1        4     Y   N    S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
TREEID          S    2          1      TREE TABLE ID
GRPID           S    2          3      BASIC TASK TABLE ID
DESC            S   30          5      BASIC TASK TABLE NAME

**** NOTES ****
THE FIELDS 'TREEID' AND 'GRPID' ARE COMBINED TO FORM THE KEY FIELD.
```

```
FILE NAME: DES-TRWD.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 34
PAGE SIZE   : 512
RECORD NAME : DESDAT

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     KEY          1        4      Y   N   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
TREEID          S    2          1       TREE TABLE ID
GRPID           S    2          3       BASIC TASK TABLE ID
DESC            S   30          5       TREE WIDE TABLE NAME

**** NOTES ****
THE FIELDS 'TREEID' AND 'GRPID' ARE COMBINED TO FORM THE KEY FIELD
```

```
FILE NAME: EQC-TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: EQCREC.INC
NUMBER OF KEYS:  1
RECORD SIZE : 50
PAGE SIZE   : 512
RECORD NAME : EQCREC

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     EQCCDE       1        2      N   N   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
EQCCDE          S    2          1       EQUIPMENT ID
EQCDES          S   40          3       EQUIPMENT DESCRIPTION
EQCCOST         R    4          43      DOLLERS PER HOUR
```

FILE NAME: F4C-YEAR.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: FYREC.INC
NUMBER OF KEYS: 1
RECORD SIZE : 34
PAGE SIZE : 512
RECORD NAME : FXREC

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	FXF4CB	1	7	Y	N	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
FXF4CB	S	7	1	BEGINNING F4C CODE
FXF4CE	S	7	9	ENDING F4C CODE
FXTEID	S	2	17	TREE ID TABLE
FXBTID	S	2	19	PERM. UNIT COST ID TABLE
FXSMID	S	2	21	SUMMARY ID TABLE
FXPTID	S	2	23	TEMP. UNIT COST ID TABLE
FXUNID	S	2	25	TOTAL UNIT COST ID TABLE
FXBYER	S	4	27	BEGINNING YEAR
FXEYER	S	4	31	ENDING YEAR

FILE NAME: FACTAB.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 1

FORMAT USED : A80

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
FGID	S	9	1	FACILITY ID
FGSUBI	S	2	10	SUBINSTALLATION ID
FGAREA	S	2	12	AREA ID
FGDESC	S	30	14	FACILITY DESCRIPTION
FGF4C	S	7	29	F4C CODE
FGNUM	I	2	36	NUMBER OF FACILITIES
FGZONE	I	2	39	TRAVEL ZONE
FGMTH	I	2	41	WORK PERFORMANCE METHOD
FGSCM	I	2	43	SPECIAL CONDITION MULTIPLIER ID
FGCHNG	S	8	45	LAST CHANGED DATE
FGLCAL	S	8	53	LAST CALCULATION DATE
FGFUND	S	10	61	FACILITY FUNDING PROFILE
FGCAL	I	2	63	CALCULATION MODELING ID
FGSQFT	I	4	64	FLOOR AREA (SQ FT)
FGCYR	I	2	73	CONSTRUCTION YEAR
FGSDSP	S	8	77	SCHEDULED DISPOSAL DATE

**** NOTES ****
USED TO BUILD THE 'FACILITY.XDB' FILE IN BATCH MODE.

FILE NAME: FACILITY.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: FAREC.INC
NUMBER OF KEYS: 3
RECORD SIZE : 120
PAGE SIZE : 1024
RECORD NAME : FGREC

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	FGID	1	9	N	N	S
1	FGXSEQ	11	4	N	N	S
2	FGF4C	49	7	Y	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
FGID	S	9	1	FACILITY ID
FGXSEQ	S	4	11	FACILITY SEQUENCE NUMBER
FGDESC	S	30	15	FACILITY DESCRIPTION
FGSUBI	S	2	45	SUBINSTALLATION ID
FGAREA	S	2	47	AREA ID
FGF4C	S	7	49	F4C CODE
FGNUM	I	2	57	NUMBER OF FACILITIES
FGZONE	I	2	59	TRAVEL ZONE
FGSQFT	I	4	61	FLOOR AREA (SQ FT)
FGCYR	I	2	65	CONSTRUCTION YEAR
FGSDSP	S	8	67	SCHEDUALED DISPOSAL DATE
FGFUND	S	2	75	FACILITY FUNDING PROFILE
MODSYS	S	1	77	COMPONENTS ENTERED
BFAPCT	I	2	79	PERCENTAGE OF BASE FACILITY
FGSCM	I	2	85	SPECIAL CONDITION MULTIPLIER ID
FGMTH	I	2	87	WORK PERFORMANCE METHOD
FGCAL	I	2	89	CALCULATION MODELING ID
FGCHNG	S	8	91	LAST CHANGED DATE
FGLCAL	S	8	99	LAST CALCULATION DATE

FILE NAME: FFPROF.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: FFPROF.INC
NUMBER OF KEYS: 1
RECORD SIZE : 90
PAGE SIZE : 512
RECORD NAME : FFREC

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	FFKEY	1	2	N	N	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
FFKEY	S	2	1	FACILITY FUNDING PROFILE ID
FFDSCR	S	40	3	PROFILE DESCRIPTION
FFLAP	S	(5)2	43	IN-HOUSE LABOR APPROPRIATION ID
FFLPR	I	(5)2	53	IN-HOUSE LABOR PERCENTAGE
FFEAP	S	(5)2	63	IN-HOUSE EQUIPMENT APP. ID
FFEPR	I	(5)2	73	IN-HOUSE EQUIPMENT PERCENTAGE
FFLNO	I	4	83	NUMBER OF LABOR ACCOUNTS
FFENO	I	4	87	NUMBER OF EQUIPMENT ACCOUNTS

FILE NAME: INSTINFO.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 6

**** FILE INFORMATION ****

LINE	FORMAT	FIELDS
1	A30	INSTNAM
2	1X,A4,1X,A4	BEGYR,ENDYR
3	3F5.2	MATADJ,MATTAF,RMFTAF
4	A2	ORGID
5	A2	MAXLIN
6	A1	VDRIVE

***** FIELD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	DESCRIPTION
INSTNAM	S	30	INSTALLATION NAME
BEGYR	S	4	BEGINNING REPORT YEAR
ENDYR	S	4	ENDING REPORT YEAR
MATADJ	R	4	MATERIAL LOCATION ADJUSTMENT FACTOR
MATTAF	R	4	MATERIAL TIME ADJUSTMENT FACTOR
RMFTAF	R	4	RMF TIME ADJUSTMENT FACTOR
ORGID	S	2	ORGANIZATION ID
MAXLIN	S	2	MAXIMUM NUMBER OF LINES PER PAGE
VDRIVE	S	1	DRIVE DESIGNATION OF VIRTUAL DISK, C=NONE

FILE NAME: ORGFGC.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 2
RECORD SIZE : 52
PAGE SIZE : 512
RECORD NAME : ORGREC

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	ORGKEY	1	4	N	N	S
1	MACOMID	43	2	Y	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
ORGKEY	S	4	1	ORGANIZATION ID
ORGCDE	S	2	5	ORGANIZATION CODE
INSTID	S	2	7	INSTALLATION ID
ORGDESC	S	30	13	ORG. DESCRIPTION
MACOMID	S	2	43	MACOM ID
RELCDE	S	6	45	RELATION CODE
SUBCDE	S	2	51	SUB-INSTALLATION CODE

FILE NAME: PMCOMP.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: *** NONE ***
NUMBER OF KEYS: 3
RECORD SIZE : 64
PAGE SIZE : 512
RECORD NAME : COMPDT

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	STF4C	1	8	Y	Y	S
1	CASCES	17	8	Y	Y	S
2	IFSNUM	25	8	Y	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
STF4C	S	8	1	STARTING F4C CODE
ENDF4C	S	8	9	ENDING F4C CODE
CASCES	S	8	17	TASK/COMPONENT ID
IFSNUM	S	8	25	IFS NUMBER
PMNUM	S	2	33	NUMBER OF THE CURRENT PREDICTION MODEL
LCALDTE	S	8	35	LAST CALCULATION DATE
LMRGDTE	S	8	43	LAST MERGE DATE
TOTPM	S	2	51	TOTAL NUMBER OF ALLOWABLE PRED. MODELS
ALLOWPM	S	(6)2	53	LIST OF THE ALLOWABLE PRED. MODELS

FILE NAME: PMDEF.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 42
PAGE SIZE : 512
RECORD NAME : PMDDAT

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
*****	*****	********	******	***	***	****
0	PMNUM	1	2	N	N	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
*****	****	*******	*********	************
PMID	S	2	1	PREDICTION MODEL ID
PMDEF	S	40	3	PREDICTION MODEL DEFINITION

FILE NAME: PMF4C.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 48
PAGE SIZE : 512
RECORD NAME : F4CDAT

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
*****	*****	********	******	***	***	****
0	STF4C	1	8	N	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
*****	****	*******	*********	************
STF4C	S	8	1	STARTING F4C CODE
ENDF4C	S	8	9	ENDING F4C CODE
PMNUM	S	2	17	NUMBER OF THE CURRENT PREDICTION MODEL
LCALDTE	S	8	19	LAST CALCULATION DATE
LMGRDTE	S	8	27	LAST MERGE DATE
TOTPM	S	2	35	TOTAL NUMBER OF ALLOWABLE PRED. MODELS
ALLOWPM	S	2(6)	37	LIST OF THE ALLOWBLE PRED. MODELS

```
FILE NAME: RMF-FACT.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 336
PAGE SIZE   : 512
RECORD NAME : RMFREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     STAMS       1         7      N   Y   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
STAMS           S      7        1       STARTING AMS CODE
ENDAMS          S      7        9       ENDING AMS CODE
RMCOST          R   (80)4      17       ARRAY OF COST FACTORS (80 YEARS)
```

```
FILE NAME: RSMTTOTL.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: RSREC.INC
NUMBER OF KEYS:  1
RECORD SIZE : 340
PAGE SIZE   : 1024
RECORD NAME : RSREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     RSRTSK      1         7      N   N   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
RSRTSK          S       8        1       COMPONENT ID
RSRTRD          I       2        9       TRADE INDEX
RSRWPM          I       2       11       WORK PERFORMANCE METHOD
RSRBFY          I       2       13       BEGINNING YEAR
RSRNFY          I       2       15       NUMBER OF YEARS
RSROCC          R    (10)4      17       NUMBER OF OCCURRENCES (10 YEARS)
RSRTOT          R    (10)4      57       TOTAL COSTS (10 YEARS)
RSRHRS          R   (3,10)      97       ARRAY OF HOURS (10 YEARS)
RSRDLR          R   (3,10)     217       ARRAY OF COSTS (10 YEARS)

**** NOTES ****
BOTH TWO DIMENTIONAL ARRAYS (RSRHRS & RSRDLR) ARE DIVIDED AS FOLLOWS:
  (1,I) = LABOR
  (2,I) = EQUIPMENT
  (3,I) = MATERIALS
```

FILE NAME: SCMDEF.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS: 1
RECORD SIZE : 32
PAGE SIZE : 512
RECORD NAME : SCMDAT

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
*****	*****	********	******	***	***	****
0	SCMID	1	2	N	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
*****	****	*******	*********	************
SCMID	S	2	1	SPECIAL CONDITION MULTIPLIER ID
SCMDEF	S	30	3	SCM DEFINITION

FILE NAME: SCMDEF.DAT
FILE TYPE: TEXT

FORMAT USED: A2,A30

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
*****	****	*******	*********	************
CODE	S	2	1	SPECIAL CONDITION MULTIPLIER ID
DESC	S	30	3	SCM DEFINITION

```
FILE NAME: SCMIDxx.DAT
FILE TYPE: TEXT
NUMBER OF LINES: 20

**** FILE INFORMATION ****
LINE    FORMAT       FIELDS
*****  ********     ******
 1        I5         NUMBER OF DATA LINES(19)
 2-20  A7,3X,F10.2   COMPONENT ID, CALCULATION TOTALS

***** RECORD INFORMATION *****
LINE    COMPONENT  CALCULATION
*****  ********** ************
 2       0000000   F(9)*F(13)
 3       0300000   F(1)*F(2)*F(5)*F(6)*F(7)*F(8)
 4       0400000   F(1)*F(2)*F(5)*F(6)*F(8)*F(11)
 5       0410000   F(15)
 6       0415400   F(4)
 7       0415500   F(4)
 8       0415800   F(4)
 9       0415900   F(4)
10       0415A00   F(4)
11       0415E00   F(14)
12       0415F00   F(14)
13       0415G00   F(14)
14       0415H00   F(14)
15       0420000   F(14)
16       0430000   F(14)
17       0500000   F(11)
18       0530000   F(12)
19       0540000   F(12)
20       0600000   F(11)

**** NOTES ****
THE 'xx' IS REPLACES BY THE SPECIAL CONDITION MULTIPLIER ID
```

```
FILE NAME: STDREP.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 4
PAGE SIZE   : 512
RECORD NAME : STDREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     STDKEY      1        4      N   N   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME                (BYTES)
*****          **** ******* *********  ************
STDAPR          S   2          1       APPROPRIATION ID
STDAMS          S   2          3       AMS CODE

**** NOTES ****
 THE FIELDS 'STDAPR' AND 'STDAMS' ARE COMBINED TO FORM THE KEY FIELD
```

```
FILE NAME: SUB_TAB.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: ***NONE***
NUMBER OF KEYS:  1
RECORD SIZE : 32
PAGE SIZE   : 512
RECORD NAME : ARDAT

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     CODE         1         2      N   Y   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME               (BYTES)
*****          **** ******* *********  ************
CODE            S       2          1    SUBINSTALLATION ID
DEF             S      30          3    SUBINSTALLATION DESCRIPTION
```

```
FILE NAME: UNCDSC.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: UNCDSC.INC
NUMBER OF KEYS:  1
RECORD SIZE : 52
PAGE SIZE   : 512
RECORD NAME : UNCRC

**** KEY INFORMATION ****
KEYNO  FIELD     POSITION  LENGTH  DUP MOD TYPE
*****  *****     ********  ******  *** *** ****
 0     UNTID        1         2      N   Y   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME               (BYTES)
*****          **** ******* *********  ************
UNTID           S       2          1    UNIT COST TABLE ID
DESP            S      50          3    DESCRIPTION
```

```
FILE NAME: UNC-FACT.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: UXREC.INC
NUMBER OF KEYS:  1
RECORD SIZE : 360
PAGE SIZE   : 512
RECORD NAME : UNCREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     UNCID       1         2      N   Y   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME               (BYTES)
*****          **** ******* *********  ************
UNCID           S      2         1      UNIT COST TABLE ID
UNCSPM          R      4         5      UNIT COST SPECIAL CONDITION MULTIPLIER
UNARM           R      4         9      ANNUAL RECURRING MAINTAINANCE FACTOR
UNCOST          R   4(80)       13      UNIT COST FACTORS FOR 80 YEARS
```

```
FILE NAME: URR.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: URRINC.INC
NUMBER OF KEYS:  1
RECORD SIZE : 90
PAGE SIZE   : 512
RECORD NAME : URRREC

**** KEY INFORMATION ****
KEYNO  FIELD    POSITION  LENGTH  DUP MOD TYPE
*****  *****    ********  ******  *** *** ****
 0     AMSCOD      1         10     N   Y   S

***** RECORD INFORMATION *****
FIELD          TYPE LENGTH  POSITION   DESCRIPTION
NAME               (BYTES)
*****          **** ******* *********  ************
AMSCOD          S      10        1      AMS CODE
URRKSF          R    4(10)       11     URR KILO SQ FT
URRKDO          R    4(10)       51     URR THOUSAND DOLLAR
```

FILE NAME: URRAPR.XDB
FILE TYPE: BTRIEVE
INCLUDE FILE: NONE
NUMBER OF KEYS: 1
RECORD SIZE : 50
PAGE SIZE : 512
RECORD NAME : APRMUL

**** KEY INFORMATION ****

KEYNO	FIELD	POSITION	LENGTH	DUP	MOD	TYPE
0	AMSCOD	1	6	N	Y	S

***** RECORD INFORMATION *****

FIELD NAME	TYPE	LENGTH (BYTES)	POSITION	DESCRIPTION
APRID	S	6	1	APPROPRIATION ID
MULPLY	R	4(10)	7	MULTIPLIER

6 STANDARD PROGRAMMING PACKAGES

These programming packages are used by the MRPM system:

1. MSFORTRAN 3.31
2. BTRIEVE
3. MS CHART

Each package has its own printed documentation.

7 REQUIRED PROGRAMMING PRACTICES

To ensure system uniformity, programmers should follow these instructions:

1. Always get source codes from and return them to the coordinator; do not update the execution file in the main machine. Only the coordinator is allowed to write into the main machine.

2. Test the program as you work on it, and make sure it is 100 percent error free. Write down all testing procedures for future reference.

3. The first line of each screen should be a header that contains the exact words of the selection menu.

4. Always show the computer status on the screen: 'COMMAND MODE', 'EDIT MODE','ADD MODE'; this helps users know where they are.

5. When the user presses 'F8' to delete records, always ask to make sure the data should be deleted.

6. Use the bottom line for standard functions. Show only functions that can be used in this mode.

7. On reports and lists to the printer, be sure to print the last page.

8. Use the 'F6 BEGIN' key to start the processing on all functions.

9. Programs must return to the previous screen after successful execution.

10. Use the standard error message subroutine "ERRMSG" to handle all the pop-up window style error messages.

11. Always make a source code backup before working on the source code. Always save the final program on a diskette in your diskette file. Write the date on the diskette.

12. There is no 'STOP' command in the program.

13. Increase labels in increments of 100.

Figure 1 shows an example of a well-documented program.

```
      PROGRAM RECFACZ
* ==> this program recovers the general facility information
*     stored in FACILITY.XDB from subdirectories
      IMPLICIT    INTEGER*2 (A-Z)
      CHARACTER   SDIR*2,EDIR*2,BUFFER*30
      INTEGER*2   FILE1(64),FILE2(64),NSDIR,NEDIR,DIR
      INTEGER*2   CODE,OP
      CHARACTER   KEY*9
      CHARACTER   PATH*18,SDRIVE*1,NEWDIR*2
      LOGICAL*2   THERE,OPEND,SUCESS
      COMMON /COLORS/ COLOR
* ==> include file for error message
$INCLUDE:'BTEROR.INC'
* ==> include file for FACILITY.XDB
$INCLUDE:'FAREC.INC'
* ==> include files for color table
$INCLUDE:'COLOR1.INC'
$INCLUDE:'COLOR2.INC'
      ROUTIN = 'REC-FAC'
* ==> initialize the error message subroutine
      CALL ERRMSG(-1,ROUTIN,'INITAL')
* ==> install function keys
      CALL INFKEY
* ==> Make sure Record Manager is loaded
      IF (BTREEX(0).NE.0) THEN
        INFOE(1)='Record Manager is not Loaded'
        CALL ERRMSG(99,ROUTIN,' ')
        GOTO 999
      ENDIF
* ==> set up for screen
100   CALL SCROLL (00,00,24,80,00,COLOR(2))
      CALL FLINE
      CALL FKEYS (1,1,1,1,1,1,1,1,1,2)
      CALL BOX(1,1,3,80,2,COLOR(2))
      CALL BOX(4,1,24,80,2,COLOR(2))
      CALL WT (2,25,' REBUILD FACILITY.XDB FROM SUBS    ',31,COLOR(14))
      CALL WT (10,15,'Starting Subdirectory :',23,COLOR(14))
      CALL WT (12,15,'Ending   Subdirectory :',23,COLOR(14))
* ==> read screen
200   CODE = 5
      CALL RT (10,39,SDIR,2,CODE,COLOR(8),COLOR(15))
      IF (CODE.EQ.68) GOTO 999
      IF (CODE.EQ.80.OR.CODE.EQ.13) GOTO 300
      GOTO 200
300   CODE = 5
      CALL RT (12,39,EDIR,2,CODE,COLOR(8),COLOR(15))
      IF (CODE.EQ.68) GOTO 999
      IF (CODE.EQ.64) GOTO 400
      IF (CODE.EQ.72) GOTO 200
      IF (CODE.EQ.13) THEN
        CALL FLINE
        CALL FKEYS (1,1,1,1,1,2,1,1,1,2)
      ENDIF
      GOTO 300
400   CONTINUE
* ==> create FACTEMP.XDB in current directory
      CALL CRFAC(FILE1,SUCESS)
* ==> if not created sucessfully terminate the program
      IF (.NOT.SUCESS) GOTO 999
* ==> open FACTEMP.XDB in current directory
      CALL BTREE (FILE1,0,0,'FACTEMP.XDB',9,FGREC,120,STS)
      IF (STS.NE.0) THEN
         INFOE(1) = 'ERROR OPEN'
         CALL ERRMSG(STS,ROUTIN,'DF..98')
         GOTO 999
      ENDIF
```

Figure 1. Example of a well-documented program.

```
      READ(SDIR,'(I2)') NSDIR
      READ(EDIR,'(I2)') NEDIR
      DIR = NSDIR
500   CONTINUE
      OPEND=.FALSE.
      WRITE(NEWDIR,'(I2.2)') DIR
      CALL FINDDR(NEWDIR,SDRIVE)
* ==> set up filename for subdirectory general facility file
      PATH='E:\\01\\FACILITY.XDB'c
      PATH(1:1)=SDRIVE
      PATH(4:5)=NEWDIR
* ==> open FACILITY.XDB in subdirectory
      CALL BTREE (FILE2,0,0,PATH,18,FGREC,120,STS)
      IF (STS.eq.12) THEN
         INFOE(1) = 'FACILITY.XDB not found in Subdirectory'
         INFOE(2) = PATH
         INFOA(1) = 'Please check files in Subdirectory'
         CALL ERRMSG(99,ROUTIN,' ')
         GOTO 800
      ENDIF
      IF (STS.NE.0)  THEN
         FILNAM = 'FACILITY.XDB'
         INFOE(1) = 'Error in opening file'
         CALL ERRMSG(STS,ROUTIN,'OPEN02')
         GOTO 800
      ENDIF
      OPEND=.TRUE.
* ==> write processing information to screen
      BUFFER = 'RECOVER FACILITY IN  E:\'
      BUFFER(22:22) = SDRIVE
      WRITE(BUFFER(25:26),'(I2.2)') DIR
      CALL WT (20,15,BUFFER,30,COLOR(14))
* ==> loop for copy facility records
      OP = 12
600   CONTINUE
      CALL BTREE (FILE2,OP,0,KEY,9,FGREC,120,STS)
      IF (STS.EQ.9) GOTO 700
      OP = 6
      WRITE(FGDIR,'(I2.2)') DIR
      CALL BTREE (FILE1,2,0,KEY,9,FGREC,120,STS)
      IF (STS.EQ.5) THEN
         INFOE(1) = 'Facility ID = '
         INFOE(1)(15:23) = FGID
         INFOE(2) = 'Facility ID already exists'
         INFOA(1) = 'Please check facility ID in subdirectory'
         CALL ERRMSG(99,ROUTIN,' ')
         GOTO 600
      ENDIF
      IF (STS.NE.0) THEN
         INFOE(1) = 'ERROR INSERTING FILE'
         CALL ERRMSG(STS,ROUTIN,'D..HGL')
         GOTO 990
      ENDIF
      GOTO 600
700   CONTINUE
* ==> close FACILITY.XDB in subdirectory
      CALL BTREE (FILE2,1,0,KEY,9,FGREC,120,STS)
      IF(STS.EQ.0) OPEND=.FALSE.
800   CONTINUE
      DIR = DIR + 1
      IF (DIR.GT.NEDIR) GOTO 990
      GOTO 500
990   CONTINUE
```

Figure 1. Cont'd.

```
      IF(OPEND) CALL BTREE(FILE2,1,0,KEY,9,FGREC,120,STS)
      CALL BTREE (FILE1,1,0,KEY,9,FGREC,120,STS)
* ==> overwrite FACILITY.XDB
      CALL COMMAND('COPY FACTEMP.XDB FACILITY.XDB >NUL'C,ERR)
      CALL COMMAND('DEL FACTEMP.XDB >NUL'C,ERR)
999   END
* ==> subroutine for creating general facility information
      SUBROUTINE CRFAC(FILE1,SUCESS)
      IMPLICIT INTEGER*2 (A-Z)
      INTEGER*2 FILE1(64),MAKE(32)
      LOGICAL SUCESS
$INCLUDE:'BTEROR.INC'
* ==> file information for FACILITY.XDB
      DATA MAKE/120,1024,3,5*0,1,9,0,5*0,11,4,2,5*0,49,7,3,5*0/
      SUCESS=.TRUE.
      ROUTIN = CRFAC
      CALL BTREE (FILE1,14,0,'FACTEMP.XDB',10,MAKE,64,STS)
      IF (STS.NE.0) THEN
         INFOE(1) = 'ERROR IN CREATE FILE'
         CALL ERRMSG(STS,ROUTIN,'X..987')
         SUCESS=.FALSE.
      ENDIF
      RETURN
      END
```

Figure 1. Cont'd.

8 MANAGEMENT PROCEDURES

A toll-free 800 number should be provided to all users. This hotline should be used to report possible improvements, questions, and system failures. The success of the MRPM system depends solely on how well system operators/programmers respond to the questions and problems that users bring to the telephone hotline. The following procedure will ensure that users receive a fast response to all problems and questions:

1. The hotline operator will note the problem and take immediate action to answer questions. The names of the MRPM user and the hotline operator, along with the message and any action taken, should be recorded on a sequentially numbered, three-part report log (Figure 2). When a specific function is being addressed, the function name should also be recorded. The log contains a white, yellow, and red copy of the report. The last (red) copy should be entered into the official logbook immediately. The white and yellow copies are given to the processor.

2. The processor should take action on the problem as soon as possible and record any further action taken on the report log. The processor keeps the yellow copy of the report for a record, and passes the white copy to the reviewer.

3. If a system command causes the question or problem, then a reviewer should reassess the complete command to ensure that the command works as intended. This second check is for the sake of quality assurance.

4. The problem log is returned to the hotline operator, who will call the users to report the action on their specific problems. It is important to maintain this direct contact with the user, and to keep an informal relationship between the system operator and the user. When the user's problem has been resolved, the completed white form should replace the red action form in the official report log book.

5. At any step in the process, emergencies should be referred to the supervisor for review. Periodically, a supervisor should review the accumulation of reports to prioritize the problems recorded in the log, to specify action to resolve specific problems, and to delegate the workload.

6. A regular newsletter should be mailed to each user, giving a short description of new changes to the system created in response to user requests. Users and their organizations should be credited for their suggestions.

Maintenance Resource Prediction Model

Report Log Number: _____

Report: Reporters Name/Org: _____ Tel. No.: _____

Date: _____ Time: _____ Received by: _____

Message: _____

_____

_____

_____

Action taken: _____

_____

Related commands: _____

Review: Reviewed by: _____ Date: _____

Priority: _____ Sent to: _____ Date sent: _____

Comments: _____

_____

_____

_____

_____

Process: Processed by: _____ Completion date: _____

Action taken: _____

_____

_____

_____

_____

Review: Reviewed by: _____ Date: _____

Comments: _____

_____

_____

Approval: _____

Feedback: Date reporter called: _____

Callers name: _____

_____

Figure 2. MRPM report log.

9 RESOURCES

There are five basic functions that must be maintained to provide full-service support:

1. Supervision
2. Training
3. Hotline
4. HQ-IFS System Maintenance
5. HQ-IFS Operations.

Supervision

The functions of the supervisor include scheduling training; review and assignment of report logs; management of all corrections, improvements, and problem identification. This function will probably consume 25 percent of a man-year at a GS-11 level, at a cost of approximately $12,800 per year.

Training

There are three types of training:

1. Self-teaching using the system manual
2. On-site training of the user
3. Centralized (group) training.

The self-teach method requires the user to have access to a resource person to answer questions at later stages of progress. This function is performed by the hotline operator.

On-site training entails sending one person to a site for a minimum of 3 days. The user provides equipment and training room. The cost for each such session would be:

GS-11 Trainer 5 days @ $200/day	$1000
TDY, 4 days @$100/day, air fare and car	1000
Supplies, manuals, etc.	300
Total	$2300

This training method is by far the best possible training situation for both the trainer and the students. Five classes per year at $2300 each would total $11,500.

Centralized training is the most expensive way to perform training in the Army. All students must travel to one central site. The central site must rent computer equipment to perform the training. (During this training each student should have a PC and no more than two students should be assigned to one PC.) This equipment may not be the same as that used by the trainees at their installations. A central Army training center must be paid to plan and conduct the training. Estimated costs will be:

GS-11 Trainer 5 days @ $200/day	$1000	
TDY, 4 days @$100/day, air fare and car	1000	
Supplies, manual, etc.	300	
Computer rental (if available) $100/day @ 5 days	500	each
Room rental $100/day @ 3 days	300	each
Student TDY, 4 days @$100/day, air fare and car	800	each

For a class of 20 students (10 computers) Total $27,300

Hotline

The hotline is a telephone number used to answer user questions, handle user problems, and report suggestions for improvement. This number should be given to all users. The person answering the hotline should be able to either answer all basic problems or refer the request to someone else for action. This activity would probably consume about 15 percent of one GS-9 or $8000 per year.

PC System Maintenance

One standard system will be required at an initial purchase cost of $35,000. Annual system hardware maintenance costs would run $2000/yr. The U.S. Army Construction Engineering Research Laboratory (USACERL) will transfer the system to the U.S. Army Engineering and Housing Support Center (USAEHSC) at a cost of $20,000. Two GS-9 Fortran programmers must be trained on the use of the system. Training will take 6 months at a cost of $30,000. Normal annual requirement will be the equivalent of one half-time person at a cost of $15,000/yr.

Newsletter

A quarterly newsletter should inform users of updates and answers to common questions. The annual cost is estimated at $5000.

Cost Summary

Initial Costs		*Annual Costs*	
USACERL transfer	$20K	Supervisor (1/4 time)	$12K
Personal computer	35K	User training (five classes onsite)	12K
Fortran program (2 @6 mo)	30K	Hotline (15 percent)	8K
		PC maintenance	2K
Total	$85K	Fortran programmer (1/2 time)	15K
		Newsletter	5K
		Basic supplies	4K
		Total	$58K

USACERL DISTRIBUTION

Chief of Engineers
ATTN: CEHEC-IM-LH (2)
ATTN: CEHEC-IM-LP (2)
ATTN: CERD-L
ATTN: DAEN-ZCP-B

CEHSC 22060
ATTN: CEHSC-FM-R

Ft Belvoir, VA
ATTN: CECC-R 22060

Defense Technical Info. Center 22304
ATTN: DTIC-FAB (2)

10
8/91